ANALISIS PERBANDINGAN METODE LSTM DAN BiLSTM UNTUK KLASIFIKASI SINYAL JANTUNG *PHONOCARDIOGRAM*

TUGAS AKHIR

Program Studi
S1 TEKNIK KOMPUTER

Oleh:
Muhammad Gerald Rizky
17410200021

FAKULTAS TEKNOLOGI DAN INFORMATIKA
UNIVERSITAS DINAMIKA
2021

ANALISIS PERBANDINGAN METODE LSTM DAN BiLSTM UNTUK KLASIFIKASI SINYAL JANTUNG *PHONOCARDIOGRAM*

TUGAS AKHIR

Diajukan sebagai salah satuu syarat untuk menyelesaikan
Program Sarjana Teknik

UNIVERSITAS Dinamika

Oleh:

Nama	**: Muhammad Gerald Rizky**
NIM	**: 17410200021**
Program Studi	**: S1 Teknik Komputer**

FAKULTAS TEKNOLOGI DAN INFORMATIKA
UNIVERSITAS DINAMIKA
2021

TUGAS AKHIR

ANALISIS PERBANDINGAN METODE LSTM DAN BiLSTM UNTUK KLASIFIKASI SINYAL JANTUNG *PHONOCARDIOGRAM*

Dipersiapkan dan disusun oleh

Muhammad Gerald Rizky

NIM: 17410200021

Telah diperiksa, diuji dan disetujui oleh Dewan Penguji

Pada: Agustus 2021

Susunan Dewan Pembahas

Pembimbing:

I. Dr. Jusak

NIDN 0708017101

II. Ira Puspasari, S.Si., M.T.

NIDN 0710078601

Digitally signed by Universitas Dinamika
Date: 2021.08.24 18:53:27 +08'00'

Digitally signed by Universitas Dinamika
Date: 2021.08.24 14:26:55 +07'00'

Pembahas:

I. Pauladie Susanto, S.Kom., M.T.

NIDN 0729047501

Digitally signed by Universitas Dinamika
Date: 2021.08.25 09:13:40 +07'00'

Tugas Akhir ini telah diterima sebagai salah satu persyaratan

Untuk memperoleh gelar Sarjana

Digitally signed by Universitas Dinamika
Date: 2021.08.25 16:08:25 +07'00'

Tri Sagirani, S.Kom., M.MT.

NIDN: 0731017601

Dekan Fakultas Teknologi dan Informatika

UNIVERSITAS DINAMIKA

"Kurang cerdas dapat diperbaiki dengan belajar. Kurang cakap dapat dihilangkan dengan pengalaman. Namun tidak jujur itu sulit diperbaiki."

– Bung Hatta

UNIVERSITAS Dinamika

"Kupersembahkan tugas akhir ini kepada kedua orang tua dan teman-teman serta yang selalu memberikan dukungan serta semangat."

UNIVERSITAS Dinamika

SURAT PERNYATAAN
PERSETUJUAN PUBLIKASI DAN KEASLIAN KARYA ILMIAH

Sebagai mahasiswa Universitas Dinamika, saya :

Nama	: Muhammad Gerald Rizky
NIM	: 17.41020.0021
Program Studi	: S1 Teknik Komputer
Fakultas	: Fakultas Teknologi dan Informatika
Jenis Karya	: Laporan Tugas Akhir
Judul Karya	: **ANALISIS PERBANDINGAN METODE LSTM DAN BiLSTM UNTUK KLASIFIKASI SINYAL JANTUNG PHONOCARDIOGRAM**

Menyatakan dengan sesungguhnya bahwa:

1. Demi pengembangan Ilmu Pengetahuan, Teknologi dan Seni, saya menyetujui memberikan kepada Universitas Dinamika Hak Bebas Royalti Non-Eksklusif (*Non-Exclusive Royalti Free Right*) atas seluruh isi/ sebagian karya ilmiah saya tersebut di atas untuk disimpan, dialihmediakan dan dikelola dalam bentuk pangkalan data *(database)* untuk selanjutnya didistribusikan atau dipublikasikan demi kepentingan akademis dengan tetap mencantumkan nama saya sebagai penulis atau sebagai pemilik pencipta dan Hak Cipta

2. Karya tersebut di atas adalah karya asli saya, bukan plagiat baik sebagian maupun keseluruhan. Kutipan, karya atau pendapat orang lain yang ada dalam karya ilmiah ini adalah semata hanya rujukan yang dicantumkan dalam Daftar Pustaka saya

3. Apabila dikemudian hari ditemukan dan terbukti terdapat tindakan plagiat pada karya ilmiah ini, maka saya bersedia untuk menerima pencabutan terhadap gelar kesarjanaan yang telah diberikan kepada saya.

Demikian surat pernyataan ini saya buat dengan sebenarnya.

Surabaya, 22 Juni 2021

Yang menyatakan

Muhammad Gerald Rizky

NIM : 17.41020.0021

ABSTRAK

Penyakit jantung salah satu penyakit yang mengancam kesehatan manusia. Penyakit penyebab kematian yang paling tinggi di kalangan masyarakat adalah jantung koroner. Penyakit jantung menjadi penyakit yang menempati posisi puncak di dunia dan di negara Indonesia sendiri. Penyakit jantung dapat dideteksi semenjak dini melalui klasifikasi sinyal suara jantung normal dan tidak normal. Pada penelitian telah dilakukan proses klasifikasi menggunakan metode *Long Short Term Memory* dan *Bidirectional Long Short Term Memory*. Tahapan awal sebelum menerapkan dua metode tersebut adalah *preprocessing* berupa normalisasi, segmentasi, dan *fast fourier transform*. Proses pelatihan metode LSTM dan BiLSTM dilakukan empat kali pengujian dengan jumlah 5 *hidden layer*. Hasil tertinggi dari pengujian yang sudah dilakukan sebanyak 500 iterasi, yaitu 81% untuk metode LSTM dan 89% untuk metode BiLSTM.

Kata Kunci : *Long Short Term Memory, Bidirectional Long Short Term Memory, Phonocardiogram, Neural Network, Deep Learning*.

UNIVERSITAS Dinamika

KATA PENGANTAR

Puji syukur atas kehadirat Allah SWT karena atas rahmat dan karunia-Nya, penulis dapat menyelesaikan laporan penelitian tugas akhir yang berjudul "Analisis Perbandingan Metode LSTM dan BiLSTM Untuk Klasifikasi Sinyal Jantung Phonocardiogram".

Pada kesempatan ini penulis mengucapkan terima kasih kepada yang sudah mendukung untuk mengerjakan tugas akhir:

1. Allah SWT karena dengan rahmatnya dan hidayahnya sehingga penulis dapat menyelesaikan penelitian Tugas Akhir ini.
2. Orang Tua dan Seluruh Keluarga tercinta yang telah memberikan dorongan dan bantuan baik moral maupun materi sehingga penulis dapat menempuh tugas akhir dan dapat menyusun laporan penelitian Tugas Akhir.
3. Bapak Dr. Jusak, selaku dosen pembimbing pertama yang telah membantu setiap kegiatan tugas akhir sehingga pelaksanaan Tugas Akhir ini.
4. Ibu Ira Puspasari, S.Si M.T., selaku dosen pembimbing kedua yang memberikan masukan serta koreksi bagi penulis selama pengerjaan laporan.
5. Bapak Pauladie Susanto, S.Kom., M.T. selaku dosen pembahas yang telah memberikan saran agar Tugas Akhir ini berjalan dengan baik.
6. Seluruh dosen Pengajar Program Studi S1 Sistem Komputer yang telah mendidik di Universitas Dinamika.
7. Seluruh pihak yang tidak dapat penulis tuliskan satu persatu yang telah membantu penulis secara langsung maupun tidak langsung.

Saran dan kritik sangat diharapkan oleh penulis untuk memperbaiki kekurangan. Mohon maaf jika terdapat kata-kata yang salah serta menyinggung perasaan pembaca. Terakhir penulis ucapkan banyak terimakasih untuk yang sudah membaca laporan akhir ini, semoga tulisan ini dapat bermanfaat bagi para pembaca.

Surabaya, 27 Juli 2021

Penulis

DAFTAR ISI

Halaman

UNIVERSITAS Dinamika

UNIVERSITAS Dinamika

DAFTAR GAMBAR

Halaman

UNIVERSITAS Dinamika

UNIVERSITAS Dinamika

DAFTAR TABEL

Halaman

UNIVERSITAS Dinamika

DAFTAR LAMPIRAN

Halaman

UNIVERSITAS Dinamika

BAB I
PENDAHULUAN

1.1. Latar Belakang

Dalam industri kesehatan keakuratan prediksi sebuah penyakit sangat penting dan memerlukan keputusan yang efektif dan efisien dalam mengambil suatu analisa dan prediksi suatu penyakit. Organ tubuh yang berperan penting dalam sistem peredaran darah manusia adalah jantung. Data dari *World Health Organization* (WHO) pada tahun 2016 menyatakan bahwa sebanyak 7,3 juta penduduk dunia mengidap penyakit jantung dan menyebabkan kematian. Walau penyakit jantung merupakan penyakit yang tidak menular, penyakit jantung adalah penyakit yang mematikan nomor satu di dunia.

Penyakit jantung disebabkan terjadinya keseimbangan yang terganggu antara suplai dan kebutuhan darah akibat penyumbatan pembuluh darah (Hananta & Muhammad, 2011). Penyebab paling utama penyakit jantung adalah merokok, konsumsi minumal beralkohol, fisik tidak aktif, resiko penyakit jantung bertambah dengan seiring bertambahnya usia, tekanan darah tinggi, memiliki kolesterol tinggi, dan berat badan yang tidak normal.

Teknik untuk mendengarkan suara jantung dengan menggunakan elektronik atau tradisional *stethoscope*, sebuah metode lama namun sangat efektif dalam melakukan diagnosis terhadap sejumlah penyakit kardiovaskular. Namun, hasil pemeriksaan yang didasarkan pendengaran dokter, juga menjadi kendala dalam menentukan hasil pemeriksaan jantung, karena merupakan hasil subjektifitas. Hal ini menjadikan analisis pendeteksian terhadap karakteristik sinyal suara jantung secara otomatis menjadi sangat penting untuk dilakukan agar tidak terjadi kesalahan diagnosa pada saat perekaman sinyal suara jantung.

Phonocardiogram adalah teknik penelusuran suara jantung. Suara-suara ini menandakan laju dan ritme jantung ketika memompa darah. Suara ini juga memberikan informasi tentang efektifitas pemompaan jantung.

Dengan berkembangnya teknik klasifikasi secara otomatis dengan neggunakan *machine learning* maupun *deep learning*, telah banyak upaya yang

dilakukan untuk menganalisa sinyal PCG secara otomatis seperti yang dilakukan oleh Tschanen pada tahun 2016 dengan menggunakan data pada *physionet* dengan menunjukkan hasil tingkat sensitivitas, spesifikasi dan skore sebesar 96%, 83% dan 89% (Heinzmann, 2016) Pada penelitian yang dilakukan Jusak dengan metode untuk indenfitikasi sinyal jantung pada bayi normal (Jusak, et al., 2020). Tahun 2019 telah dilakukan penelitian untuk mengekstraksi letak murmur pada sinyal suara jantung (Puspasari, et al., 2019). Penelitan dengan menggunakan algoritma CEEMD pernah dilakukan oleh Jusak untuk identifikasi secara semi otomatis pada sinyal suara jantung PCG. Penelitian tersebut menunjukkan hasil yang baik, nilai akurasi klasifikasi jantung normal 98%(Jusak, et al., 2021). Banyaknya penelitian tentang klasifikasi sinyal jantung telah dirangkum oleh Suyi Li pada tahun 2018 (S. Li, 2020).

LSTM didesain untuk mempelajari data apa yang akan dipakai maupun yang tidak dipakai. Karena LSTM mempunyai sebuah neuron yang didalamnya memiliki *gates*, setiap *gate* berfungsi untuk mengatur memori pada setiap neuron itu sendiri. Karena LSTM melakukan proses pengolahan dengan data masukan berupa data sekuensial, maka LSTM banyak digunakan untuk pemrosesan audio, teks, video, dan data time series.

Hasan (2017) juga melakukan penelitian dengan menggunakan metode LSTM untuk melakukan analisis sentimen pada situs IMDB. Peneliti menggunakan data inputan berupa teks. Penelitian tersebut menghasilkan hasil yang lebih akurat untuk melakukan analisis sentimen.

BiLSTM merupakan pengembangan dari LSTM biasa yang dapat meningkatkan peforma pada masalah klasifikasi. BiLSTM melakukan *training* data dua kali, tidak seperti LSTM biasa yang hanya melakukan *training* satu kali dari dataset. Dataset mentah dari proses pertama dimasukan lagi ke proses kedua. Ini dapat memberikan tambahan jaringan dan memperoleh hasil yang lebih cepat dan lebih lengkap.

Penelitian ini akan melakukan klasifikasi sinyal suara jantung *phonocardiogram* menggunakan metode LSTM dan BiLSTM dan menganalisa unjuk kerja kedua metode tersebut.

1.2. Perumusan Masalah

Berdasarkan latar belakang masalah diatas, maka dirumuskan permasalahan dalam tugas akhir ini adalah bagaimana melakukan klasifikasi sinyal jantung normal dan abnormal dalam bentuk PCG dengan menggunakan metode LSTM dan BiLSTM serta melakukan analisis perbandingan unjuk kerja terhadap kedua metode.

1.3. Batasan Masalah

Dalam pembuatan Tugas Akhir ini, ruang lingkup penelitian hanya akan dibatasi pada:

1. Dataset sinyal jantung berasal dari *Physionet.*
2. Jumlah sample 3240 data normal dan abnormal sinyal *phonocardiogram.*

1.4. Tujuan

Berdasarkan uraian latar belakang dan rumusan masalah di atas, maka tujuan dari tugas akhir ini yaitu sebagai berikut:

1. Melakukan klasifikasi sinyal jantung dalam bentuk PCG dengan menggunakan metode LSTM dan BiLSTM.
2. Melakukan deteksi sinyal jantung dalam bentuk PCG untuk dikategorikan sebagai sinyal jantung normal dan abnormal.
3. Membandingkan hasil LSTM dan BiLSTM.

1.5. Manfaat Penelitian

Adapun manfaat penyusunan Tugas Akhir ini untuk beberapa koelompok adalah sebagai berikut:

1. Proses pendeteksian penyakit kardiovaskular dengan lebih cepat dan efektif, sehingga pasien tidak perlu melakukan banyak tahapan untuk memonitoring keadaan jantungnya.
2. Pengambilan keputusan secara lebih objektif diharapkan dapat membantu para dokter dalam melakukan analisis terhadap pasien dengan adanya klasifikasi sinyal jantung secara otomatis.

UNIVERSITAS Dinamika

BAB II
LANDASAN TEORI

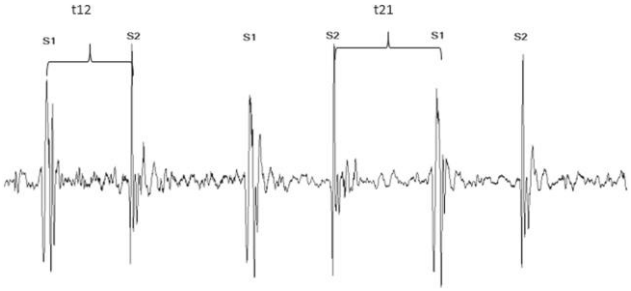
2.1 Jantung

Organ vital dalam tubuh manusia yang bertugas memompa darah sehingga mengalir keseluruh tubuh adalah jantung. Jantung terdiri dua bagian yaitu serambi pada bagian atas, dan bilik pada bagian bawah. Otot - otot pada jantung bertugas memompa darah dari satu ruangan ke ruangan lainnya. Setiap proses pemompaan, katup jantung membuka sehingga darah dapat mengalir ke ruangan yang dituju. Katup jantung akan menutup untuk mencegah aliran balik darah.

2.1.1 Suara Jantung

Saat jantung berdetak menghasilkan dua suara berbeda yang bisa didengarkan menggunakan stetoskop. Suara jantung pertama atau biasa dinayatakan dengan lub disebabkan oleh penutupan katup triscupid dan mitral yang mengalirkan darah dari serambi jantung ke bilik jantung dan mencegah aliran balik ke serambi jantung, suara jantung pertama juga biasa disebut S1. Suara jantung kedua atau biasa dinyatakan dengan dub disebut suara jantung kedua S2 disebabkan oleh penutupan katup semilunar yang memungkinkan darah bebas ke sistem sirkulasi paru-paru dan sistemik. Pada jantung tidak normal aliran darah menjadi tidak lancar dan mengakibatkan tercipta suara diluar S1 dan S2.

Saat jantung dalam kondisi normal, pada dasarnya hanya terdapat dua macam suara jantung, yaitu S1 dan S2 seperti ditunjukkan Gambar 2.1



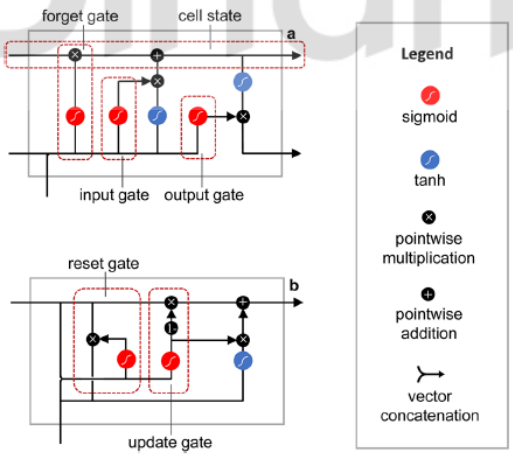
Gambar 2.1 Sinyal Suara Jantung Phonocardiogram
(Sumber: Ismail, 2018)

2.1.2 Phonocardiogram

Alat yang bisa mendengar suara jantung adalah stetoskop. *Phonocardiography* adalah suatu teknik untuk menelusuri suara jantung dan merekam getaran akustik jantung melalui mikrofon dan menampilkan sinyal pada osiloskop. Sinyal yang ditampilkan pada osiloskop adalah indikasi ritme dan laju jantung dalam memompa darah. Suara ini bisa memberikan data bagaimana jantung memompa dan katup beraktifitas secara efektif.

2.2 Long Short Term Memory Neural Network (LSTM)

Long Short – Term Memory Neural Network (LSTM) adalah salah satu arsitektur dari *Recurant Neural Network* yang telah ditingkatkan sedemikian rupa oleh Graves LSTM diusulkan sebagai pemecahan solusi terbentuknya *vanishing gradient* pada RNN dikala memproses informasi *sequential* yang panjang. Arsitektur LSTM dapat mengatasi penyimpanan memori dalam jangka waktu yang lama karena *memory cell* telah bertambah. Semua *recurrent neural network* memiliki desain rangkaian jaringan syaraf yang berulang. LSTM juga memiliki struktur yang sama namun memiliki tambahan fitur berupa gerbang pada sel.



Gambar 2.2 Struktur LSTM
(Sumber: Chung,2014)

Gate berfungsi untuk mengontrol cara keadaan internal dipertahankan atau dibuang. Struktur unit ditunjukkan pada gambar dan persamaan algoritma *input* ke *output* sel LSTM ditunjukkan pada Gambar 2.2.

LSTM akan menentukan informasi apa yang akan diabaikan dari sel. Keputusan ini dibuat oleh *forget gate layer*. *Layer* ini akan memperhatihan h_{t-1} dan x_t sehingga akan menghasilkan keluaran antara 0 dan 1. Keluaran 0 menandakan bahwa informasi akan dibuang, jika kleuaran 1 menandakan bahwa informasi tidak akan dibuang.

$$g^{(t)} = \sigma\left(b_g + U_g x^{(t)} + W_g h^{(t-1)}\right), \tag{2.1}$$

$$f^{(t)} = \sigma\left(b_f + U_f x^{(t)} + W_f h^{(t-1)}\right), \tag{2.2}$$

$$o^{(t)} = \sigma\left(b_o + U_o x^{(t)} + W_o h^{(t-1)}\right), \tag{2.3}$$

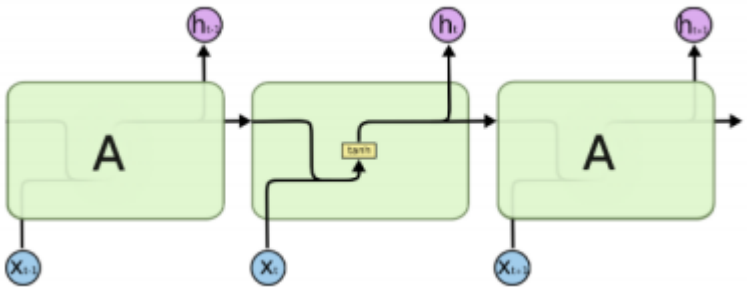
$$s^{(t)} = f^{(t)} s^{(t-1)} + g^{(t)} \sigma\left(b + U x^{(t)} + W h^{(t-1)}\right), \tag{2.4}$$

$$h^{(t)} = \tanh(s^{(t)})\, o^{(t)}, \tag{2.5}$$

Dimana σ merepresentasikan *sigmoid function* dan bobot 0-1, dan $g^{(t)}$, $f^{(t)}$, $o^{(t)}$, $s^{(t)}$ merupakan *input gate*, *forget gate*, *output gate*, dan *cell state unit.* Dan b, U, dan W merupakan *bias, input weight, dan circular weight* (Chung, 2014).

Semua LSTM layer dikoneksikan dengan *softmax function* dengan rumus fungsi:

$$softmax(x_i) = \frac{\exp(x_i)}{\sum_i \exp(x_i)}, \tag{2.6}$$



Gambar 2.3 Struktur *layer* tanh
(Sumber: Olah,2015)

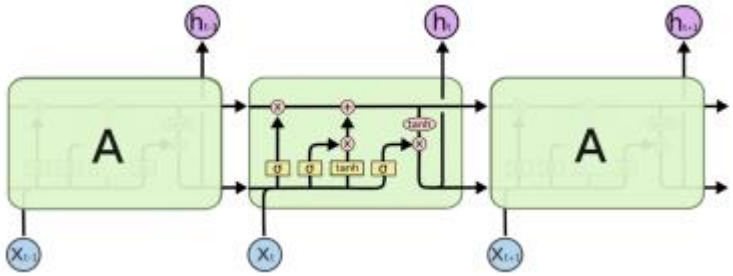
Recurrent Neural Network (RNN) proses perulangan jaringan hanya menggunakan satu layer tanh seperti pada Gambar 2.3. Persamaan tanh dijelaskan pada persamaan 2.7.

$$tanh(x) = 2\sigma(2x) - 1 \tag{2.7}$$

Dimana:

σ = fungsi aktivasi sigmoid

x = data input



Gambar 2.4 Struktur perulangan dengan empat layer pada LSTM
(Sumber: Olah,2015)

Pada struktur LSTM menggunakan empat *layer* untuk memproses modelnya seperti pada gambar 2.4. Persamaan pada perulangan LSTM menurut Hochreiter & Schmidhber (1997) dijelaskan pada persamaan 2.8.

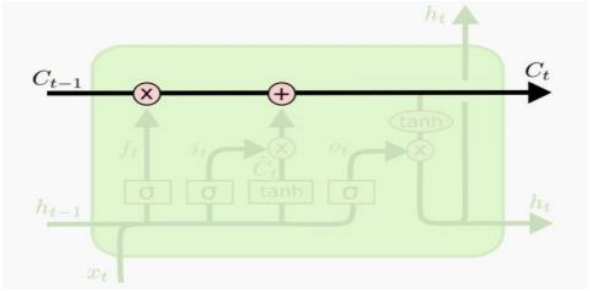
$$f_t = \sigma(W_f . \left[h_{t-1}, x_t\right] + b_f \tag{2.8}$$

$$i_t = \sigma(W_i . \left[h_{t-1}, x_t\right] + b_i$$

$$\bar{C_t} = tanh(W_c . \left[h_{t-1}, x_t\right] + b_c$$

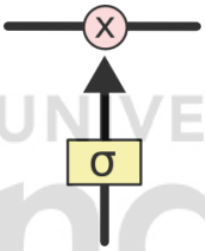
$$C_t = ft * Ct - 1 + it * C\bar{t}$$

$$o_t = \sigma(W_o . \left[h_{t-1}, x_t\right] + b_o \; ht = o_t * \tanh\,(C_t)$$



Gambar 2.5 *Cell State* pada layer LSTM
(Sumber: Olah,2015)

Hal paling utama dalam LSTM adalah *cell state*. *Cell state* merupakan garis horizontal yang menghubungkan semua *output* pada *layer* LSTM seperti pada gambar 2.5. LSTM mempunyai kelebihan untuk mengingat dan melupakan informasi dari *cell state*. Kelebihan ini disebut dengan *gates*. *Gates* terdiri dari sigmoid *layer* dan *pointwise multiplication operation* seperti pada gambar 2.6.



UNIVERSITAS Dinamika

Gambar 2.6 Sigmoid *layer*
(Sumber: Olah,2015)

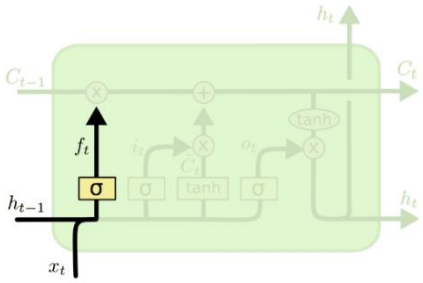
Keluaran dari sigmoid adalah angka 1 atau 0 yang menandakan apakah informasi tersebut akan dipertahankan atau dibuang. Angka 0 menandakan tidak ada informasi yang akan dipertahankan, sedangkan angka 1 menandakan semua informasi akan dipertahankan. Persamaan sigmoid dijelaskan pada persamaan 2.9.

$$\sigma(x) = 1/(1 + \in^{-x}) \quad (2.9)$$

Dimana:

$x =$ data input

$\in$ = konstanta matematika (2,71828 18284 59045 23536 02874)



Gambar 2.7 *Forget gate layer* LSTM
(Sumber: Olah,2015)

LSTM akan memutuskan informasi apa yang akan dibuang dari *cell state* yang diproses oleh sigmoid *layer*. *Forget gate layer* akan mengolah h_{t-1} dan x_t sebagai *input,* dan mengeluarkan angka 0 atau 1 pada *cell state* seperti pada Gambar 2.7. Persamaan *forget gate layer* dijelaskan pada persamaan 2.10.

$$f_t = \sigma\left(W_f.\left[h_{t-1}, x_t\right] + b_f\right) \quad (2.11)$$

Dimana:

f_t = *forget gate.*

σ = fungsi sigmoid.

W_f = nilai bobot untuk *forget gate.*

h_{t-1} = nilai keluaran sebelum orde ke t.

x_t = nilai *input* pada orde ke t.

b_f = nilai bias pada *forget gate.*

Nilai bobot dijelaskan pada persamaan 2.12

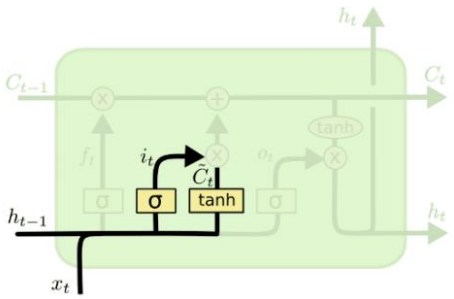
$$W = (-\frac{1}{\sqrt{\bar{d}}}, \frac{1}{\sqrt{d}} \quad (2.12)$$

Dimana:

W= bobot.

d = jumlah data.

Proses kedua adalah memutuskan informasi apa yang akan dipertahankan pada *cell state*. Pada proses ini terdapat dua bagian. Sigmoid *layer* akan memutuskan nilai mana yang akan diperbarui dan tanh *layer* membuat nilai baru yang dapat ditambahkan ke *cell state*. Proses memutuskan informasi ini digambarkan pada Gambar 2.8.



Gambar 2.8 Proses kedua metode LSTM
(Sumber: Olah,2015)

Persamaan *input gate* dijelaskan pada persamaan 2.13.

$$i_t = \sigma(W_i . \left[h_{t-1}, x_t\right] + b_i \quad (2.12)$$

Dimana:

i_t = *input gate*.

d = jumlah data.

σ = fungsi sigmoid.

W_i = nilai bobot untuk *input gate*.

h_{t-1} = nilai keluaran sebelum orde ke t.

x_t = nilai *input* pada orde ke t

b_i = nilai bias pada *input gate*.

Persamaan mendapatkan nilai baru dari tanh *layer* dijelaskan pada persamaan 2.13.

$$C_{\bar{t}} = tanh(W_c . \left[h_{t-1}, x_t\right] + b_c \quad (2.13)$$

Dimana:

$C_{\bar{t}}$ = nilai baru yang dapat ditambahkan pada *cell state*.

UNIVERSITAS Dinamika

$tanh$ = fungsi tanh.

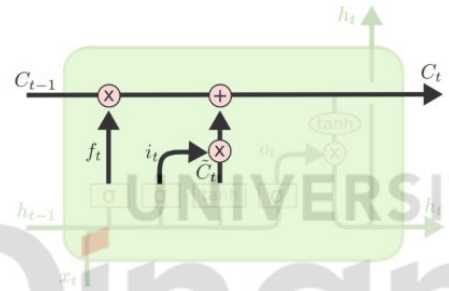
W_c = nilai bobot untuk *cell state.*

h_{t-1} = nilai keluaran sebelum orde ke t.

x_t = nilai *input* pada orde ke t

b_c = nilai bias pada *cell state.*

Proses ketiga adalah *cell state* lama ykan diperbarui menjadi *cell state* baru. Dengan cara mengkalikan f_t dengan *cell state* lama, untuk membuang informasi yang sudah ditentukan sebelumnya di *forget gate layer*. Proses diini digambarkan pada Gambar 2.9.



Gambar 2.9 Proses ketiga metode LSTM
(Sumber: Olah,2015)

Persamaan pembaruan *cell state* dijelaskan pada persamaan 2.14.

$$C_t = ft * C_{t-1} + i_t + C\bar{t} \quad (2.13)$$

Dimana:

C_t = *cell state.*

$tanh$ = fungsi tanh.

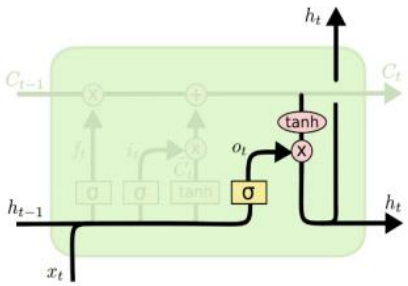
f_t = *forget gate.*

C_{t-1} = nilai *cell state* sebelum orde ke t.

i_t = nilai *input gate.*

$C\bar{t}$ = nilai baru yang ditambahkan pada *cell state.*

Proses terakhir adalah memutuskan hasil keluaran dari metode LSTM. Keluaran dan *cell state* yang telah diproses harus sesuai. *Cell state Cell state* yang menjadi keluaran adalah keputusan dari sigmoid *layer*. Keluaran dari *cell state* dimasukan pada tanh *layer* dan dikalikan dengan *gate*¸ supaya keluaran dari *cell state* sesuai dengan apa yang diputuskan sebelumnya. Proses terakhir ini digambarkan pada Gambar 2.10.



Gambar 2.10 Proses terakhir metode LSTM
(Sumber: Olah,2015)

Persamaan pada *output gate* dijelaskan pada persamaan 2.14.

$$o_t = \sigma(W_o . \left[h_{t-1}, x_t\right] + b_o \quad (2.14)$$

Dimana:

o_t = *output gate*.

σ = fungsi sigmoin.

W_o = nilai boot untuk *output gate.*

h_{t-1} = nilai keluaran sebelum orde ke t.

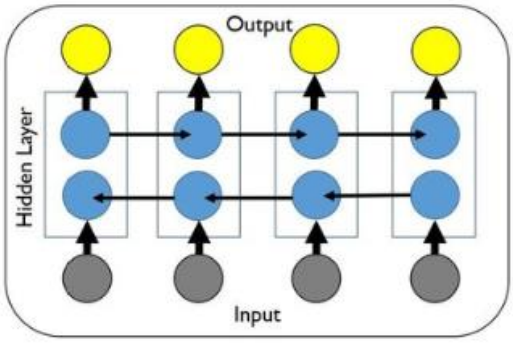
x_t = nilai *input* pada orde ke t.

b_o = nilai bias pada *output gate.*

2.3 Bidirectional Long Short Term Memory Neural Network (BiLSTM)

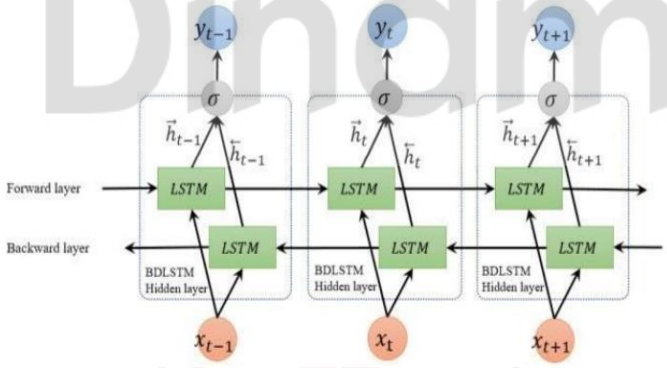
Bidirectional Long Short – Term Memory Neural Network (BiLSTM) merupakan salah satu varian dari *Long Short Term Memory* yang paling sering digunakan. Input *forward* dan input *backward* adalah 2 jenis masukan yang dimasukkan ke dalam arsitektur *Bidirectional Long Short Term Memory*. Output

dari arsitektur ini biasnya digabungkan jadi satu. Dengan layer arsitektur ini, model bisa menekuni data masa lalu *(past)* dan data masa mendatang *(future)* untuk setiap sekuen input.



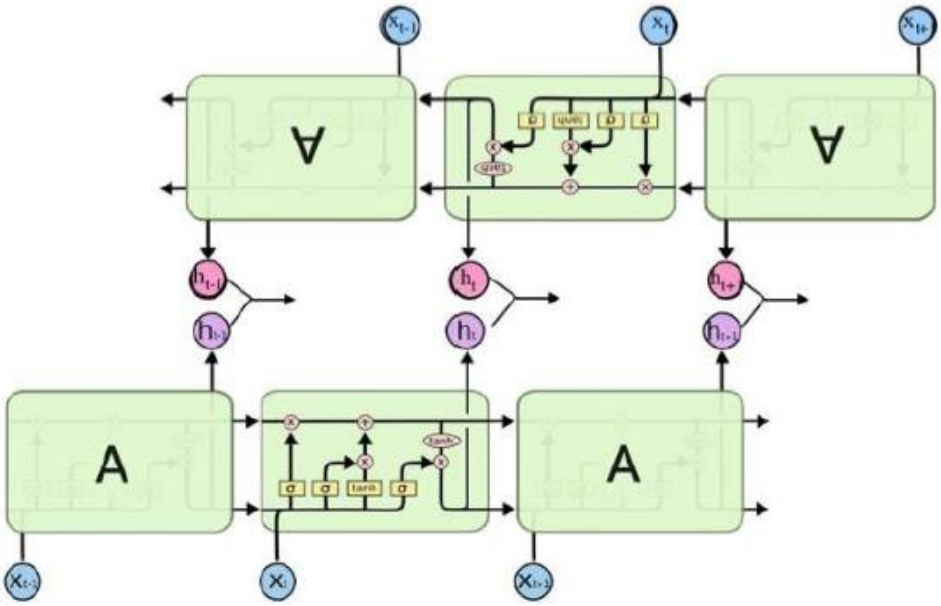
Gambar 2.11 Arsitektur BiLSTM
(Sumber: Graves & Schmidhuber 2005)

Bidirectional LSTM memanfaatkan informasi sebelumnya dan informasi setelahnya dengan memproses data dari dua arah. *Forward layer* berfungsi untuk merepresentasikan informasi sebelumnya, dan *backward layer* berfungsi untuk merepresentasikan informasi setelahnya.



Gambar 2.12 Arsitektur Hidden BiLSTM
(Sumber: gabormelli.com)

Pada Gambar 2.6 bisa dilihat dari setiap *hidden layer* keluaran unit pada *layer* bawah dan atas disatukan hingga membentuk nilai fitur yang lebih panjang dari pada LSTM biasa. Karena nilai fitur pada BiLSTM lebih panjang , maka informasi yang akan diproses pada proses selanjutnya yaitu *feed forward neural* akan mengklasifikasikan dengan lebih detail.



Gambar 2.13 Model BiLSTM
(Sumber: Varsamopoulos,2018)

Pada Gambar 2.7 adalah arsitektur dari BiLSTM yang merupakan gabungan dari dua LSTM arah maju dan arah mundur. Terdapat *hidden layer* juga yang terhubung dari LSTM maju dan LSTM mundur.

Keluaran dari LSTM dua arah *hidden layer* adalah

$$y_t = W_{\overrightarrow{hy}}\overrightarrow{h}_t + W_{\overleftarrow{hy}}\overleftarrow{h}_t \quad (2.15)$$

Dimana:

y_t = *output gate* LSTM dua arah.

$W_{\overrightarrow{hy}}$ = nilai boot untuk *output gate* LSTM maju.

$W_{\overleftarrow{hy}}$ = nilai bobot untuk *output gate* LSTM mundur.

$\overrightarrow{h}_t$ = nilai keluaran LSTM maju.

$\overleftarrow{h}_t$ = nilai keluaran LSTM mundur.

Dengan adanya *hidden layer* dua arah ini yang saling berlawanan maka model dapat memahami data dari depan dan belakang, sehingga proses pelatihan akan lebih memahami data pada *time series.* BiLSTM akan sangat berguna dalam

hal pelatihan sekuensial apabila bisa mengakses dari informasi sebelum dan sesudahnya. Jika LSTM hanya bisa mengakses informasi dari masa lalu saja, tetapi informasi masa mendatang tidak diketahui. BiLSTM bisa menjadi solusi untuk memecahkan masalah tersebut.

2.4 Library LSTM dan BiLSTM

```
tf.keras.layers.LSTM(
    units,
    activation="tanh",
    recurrent_activation="sigmoid",
    use_bias=True,
    kernel_initializer="glorot_uniform",
    recurrent_initializer="orthogonal",
    bias_initializer="zeros",
    unit_forget_bias=True,
    kernel_regularizer=None,
    recurrent_regularizer=None,
    bias_regularizer=None,
    activity_regularizer=None,
    kernel_constraint=None,
    recurrent_constraint=None,
    bias_constraint=None,
    dropout=0.0,
    recurrent_dropout=0.0,
    return_sequences=False,
    return_state=False,
    go_backwards=False,
    stateful=False,
    time_major=False,
    unroll=False,
    **kwargs
)
```

Library LSTM *layer* pada Keras dari *hardware* dan batasan *runtime* yang tersedia, *layer* ini akan memilih implementasi yang berbeda (cuDNN-based atau pure-TensorFlow) untuk memaksimalkan kinerja. Jika GPU tersedia dan semua argumen memenuhi syarat dari kernel cuDNN, *layer* akan menggunakan implementasi CuDNN yang cepat. Berikut adalah persyaratan untuk menggunakan cuDNN :

1. *activation == tanh*

2. *recurrent_activation == sigmoid*

UNIVERSITAS Dinamika

3. *recurrent_dropout* == 0

4. *unroll is false*

5. *use_bias is true*

Contohnya:

```
>>> inputs = tf.random.normal([32, 10, 8])
>>> lstm = tf.keras.layers.LSTM(4)
>>> output = lstm(inputs)
>>> print(output.shape)
    (32, 4)
>>> lstm = tf.keras.layers.LSTM(4, return_sequences=True, ret
    urn_state=True)
>>> whole_seq_output, final_memory_state, final_carry_state =
    lstm(inputs)
>>> print(whole_seq_output.shape)
    (32, 10, 4)
>>> print(final_memory_state.shape)
    (32, 4)
>>> print(final_carry_state.shape)
    (32, 4)
```

- units : berupa bilangan bulat positif.
- activation : fungsi aktivasi yang akan digunakan. Umumnya : *hyperbolic tangent* (tanh).
- recurrent_activation : fungsi aktivasi yang akan digunakan untuk langkah berulang.
- use_bias : berupa *boolean*.
- kernel_initializer : inisialisasi untuk bobot matriks kernel, digunakan untuk transformasi linier dari *input*.
- recurrect_initializer : inisialisasi untuk bobot matrix recurrent_kernel, digunakan untuk transformasi lininer dari kondisi sebelumnya.
- bias_initializer : inisialisasi untuk vektor bias.
- unit_forgot_bias : bertipe data boolean, umumnya bernilai *true*.
- kernel_reguralizer : fungsi *regularizer* diterapkan pada matriks bobot kernel.
- recurrent_regularizer : fungsi *regularizer* diterapkan pada matriks bobot recurrent_kernel.

UNIVERSITAS Dinamika

- recurrent_dropout : berupa bilangan desimal antara 0 sampai 1.
- unroll : berupa *boolean.*

```
tf.keras.layers.Bidirectional(
    layer, merge_mode="concat", weights=None, backward_layer=
    None, **kwargs
)
```

Untuk memproses data dengan tipe *time series* seperti audio, teks, video, *Bidirectional* dapat bekerja lebih baik karena memproses urutan dari awal hingga akhir dan mundur dari belakang ke depan. Contohnya, untuk memprediksi kata berikutnya dalam sebuah kalimat. Berikut adalah contoh menerapkan *bidirectional layer* :

```
model = keras.Sequential()

model.add(
    layers.Bidirectional(layers.LSTM(64, return_sequences=Tru
    e), input_shape=(5, 10)))
model.add(layers.Bidirectional(layers.LSTM(32)))
model.add(layers.Dense(10))

model.summary()
```

Bidirectional akan menyalin lapisan RNN yang akan diteruskan, dan membalik ke belakang dari lapisan baru yang disalin, sehinggi akan memproses *input* dalam urutan terbalik. Keluaran dari RNN bidirectional secara umum adalah gabungan dari keluaran lapisan maju dan keluaran lapisan mundur.

2.5 Pengujian Klasifikasi Sinyal

Tabel 2.1 Confusion Matrix

Prediksi Kelas	True Positive (TP)	False Positive (FP)
	False Negative (FN)	True Negative (TN)

Confusion matrix adalah suatu metode yang dapat digunakan untuk pengukuran, adapun penjelesasnnya, yaitu :

1. *Positive* adalah hasil yang benar.
2. *Negative* adalah hasil yang salah.
3. TP merupakan hasil dari prediksi yang positif dan sesuai dengan target yang sesungguhnya positif.
4. TN merupakan hasil dari prediksi yang negatif dan sesuai dengan target yang sesungguhnya negatif.
5. FP merupakan hasil dari prediksi yang sesungguhnya positif, namun hasil targetnya negatif.
6. FN merupakan hasil dari prediksi yang sesungguhnya negatif, namun hasil targetnya positif.

Rumus Akurasi:

$$Akurasi = \frac{TP+FN}{P+N} \quad (2.15)$$

Rumus *Precision* / *Post Predictive Value*:

$$PPV = \frac{TP}{TP+FP} \quad (2.16)$$

Rumus Recall:

$$Recall = \frac{TP}{TP+FN} \quad (2.17)$$

Rumus F1-Score:

$$F1 - Score = 2\,X\,\frac{Precision.Recall}{Precision+Recall} \quad (2.18)$$

UNIVERSITAS Dinamika

2.6 Python

Python ialah bahasa pemograman tinkat tinggi yang diciptakan oleh Guido Van Rossum pada tahun 1989 di Amsterdam, Belanda. Pada pemograman python sangatlah mudah untuk menuliskan *syntax* programnya. Python juga merupakan bahasa pemograman multiplatform yang dapat dijalankan di berbagai sistem operasi. Python dapat berjalan pada sistem operasi WINDOWS, UNIX, Linux, dan IOS. Python bahasa pemograman dengan portabilitas yang tinggi.

Bahasa pemograman python adalah jenis pemograman yang banyak diminati untuk saat ini karena keseserhanaanya. Meskipun sederhana tapi pustaka yang ada sangat lengkap misalnya modul – modul untuk keperluan jaringan, antarmuka grafis, analisis dan komputasi numerik, *Hypertext* (HTML, XML, dan lain-lain), akses *database*, dan berbagai hal lainnya. Kode bahasa pemograman python mudah untuk dibaca baik oleh yang sudah tebiasa dengan bahasa pemograman lain.

Kecenderungan penggunaan pemrograman berorientasi objek bahasa pemograman python juga sangat tepat untuk digunakan sebagai pemrograman berorientasi objek. Bahasa pemograman python disediakan gratis yang bisa didapat pada situs www.python.org.

2.7 Google Colab

Pesatnya perkembangan bahasa pemograman Python telah menarik minat Google untuk membuat *online integrated environment development* (IDE) yang disebut dengan Google Interactive Notebook atau biasa dikenal dengan Google Colab melalui situs resminya www.colab.research.google.com. Jenis lingkungan yang digunakan adalah Jupyter Notebook, dan ekstensi file adalah * .ipynb. Untuk memberi Anda informasi, Python memiliki beragam lingkungan pemrograman, dari IDLE yang telah digunakan sejak lama, hingga Spyder yang memiliki lingkungan lengkap. Namun, untuk karakter berbasis web, Google lebih memilih Notebook Jupyter. Dari sisi *software*, Google Colab telah menyiapkan sebagian besar *library* yang dibutuhkan. Dalam penelitian ini *library* yang dibutuhkan adalah Keras, TensorFlow, NumPy, Pandas dan program pendukung lainnya, misalnya untuk membuat grafik melalui Matplotlib.

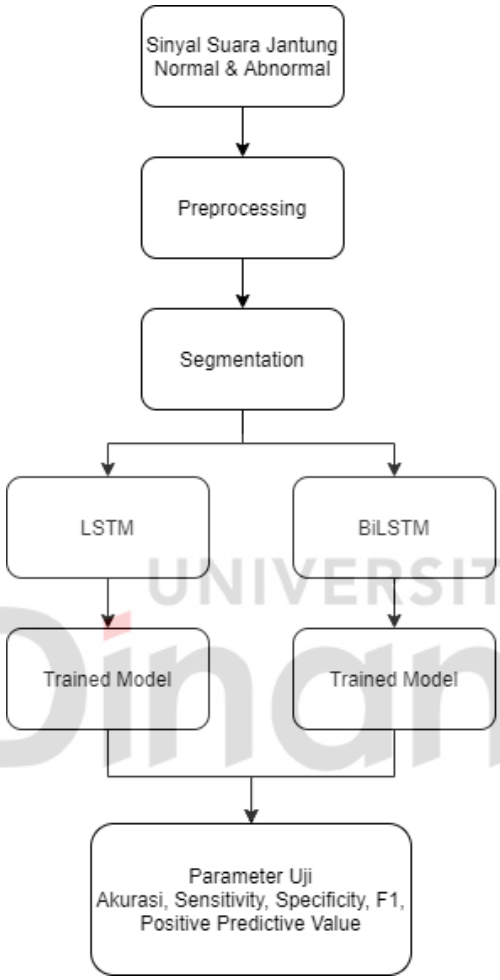
2.8 Fast Fourier Transform

Fast Fourier Transform merupakan suatu algoritma yang dapat diterapkan untuk merepresentasikan sinyal dalam domain waktu diskrit dan domain frekuensi. Algoritma transformasi *fourier* ini yang dikembangkan dari algoritma yang sudah ada sebelumnya yaitu dari *Discrete Fourier Transform* (DFT) yang sangat efisien. Proses algoritma FFT dimulai dari menguraikan sinyal dalam domain waktu pada titik N ke N sinyal domain waktu sehingga masing – masing sinyal domain terdiri satu titik saja. Selanjutnya adalah menghitung berapa N frekuensi spektrum yang berkorespondensi dengan N sinyal *domain* waktu. Proses yang terakhir yaitu spektrum N disintesis menjadi spektrum frekuensi tunggal.

UNIVERSITAS Dinamika

BAB III
METODOLOGI PENELITIAN

3.1 Metode Penelitian



Gambar 3.1 Model Perancangan

Perancangan klasifikasi sinyal jantung dilakukan menggunakan metode *Long Short Term Memory* dan *Bidirectional Long Short Term Memory* seperti yang terdapat pada gambar 3.1 terdapat beberapa tahapan dalam melakukan proses klasifikasi.

UNIVERSITAS Dinamika

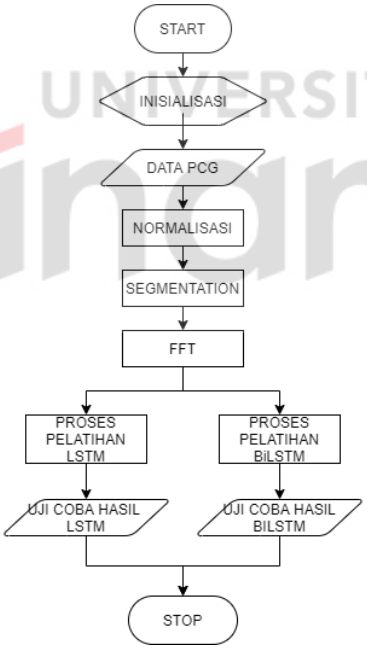
3.1.1 Pengambilan Data

Data input berupa sinyal suara jantung normal dan abnormal yang diambil dari *Physionet Cardiology Challenge* 2016. Data sinyal diambil menggunakan stetoskop digital, proses pengambilan data sangat penting karena penggunaan stetoskop digital bisa menyebabkan noise yang terekam saat proses perekaman sinyal suara jantung, karena jika terdapat noise yang terekam bisa diaanggap sebagai murmur. Data *test* dibagi menjadi 3 bagian dengan rasio 60% training, 30% validasi, dan 10% *test.*

3.1.2 Pengolahan Data

Data yang sudah diambil diberi tanda di tabel dengan angka 0 untuk sinyal normal dan angka 1 untuk sinyal tidak normal. Data training, validasi, dan *test* dipisah kedalam masing masing folder

3.1.3 Proses Pelatihan



Gambar 3.2 Flowchart Pelatihan LSTM dan BiLSTM

Dari Gambar 3.2 yang merupakan *flowchart* proses pelatihan LSTM dan BiLSTM akan dijelaskan sebagai berikut.

1. Inisialisasi

```
from scipy.io import wavfile
from pylab import*
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
from __future__ import print_function
import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Dropout, Embedding, LSTM, Bidirectional
```

Gambar 3.3 Inisialisasi Library

Inisialisasi library yang akan digunakan.

a. Scipy adalah suatu *library* untuk melakukan beberapa perhitungan *scientific* pada pemrograman *python*.

b. Pylab adalah *library* gabungan dari pyplot dan numpy untuk plotting.

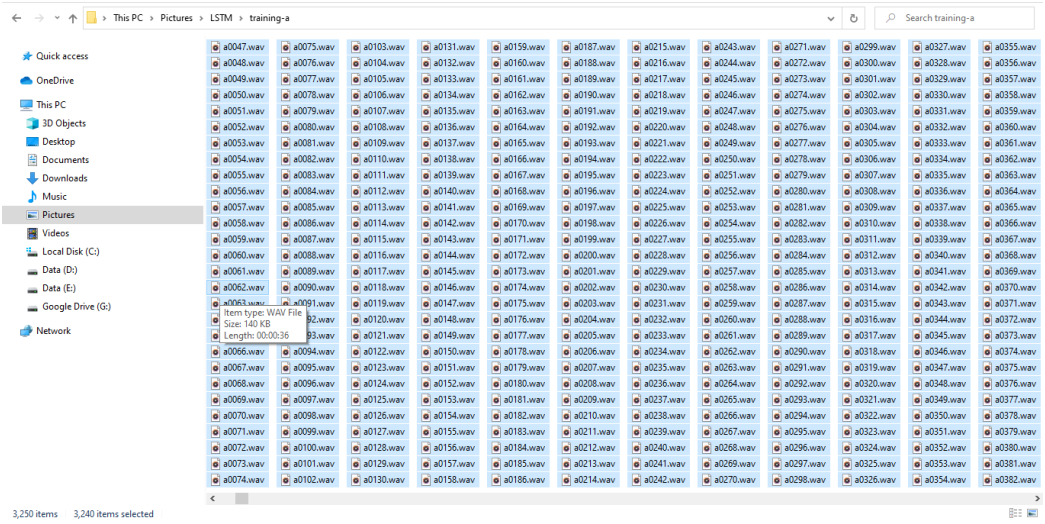
c. Pandas adalah *library* di *python* yang digunakan untuk membuat tabel, mengecek data, dan mengubah dimensi data.

d. Matplotlib adalah *library* yang dirancang agar dapat digunakan seperti Matlab, dengan menggunakan matplotlib program bisa memvisualisasikan data secara 2 dimensi maupun 3 dimensi.

e. Numpy merupakan suatu *library* yang disediakan *python* agar memudahkan operasi perhitungan tipe data nimerik, numpy menyimpan data dalam bentuk array 1 dimensi, 2 dimensi, atau lebih.

f. Keras merupakan *library* yang dibuat untuk mensederhanakan algoritma dari *library* tensorflow.

2. Data PCG



Gambar 3.4 Data sinyal suara jantung Physionet

	A	B
1	a0001	1
2	a0002	1
3	a0003	1
4	a0004	1
5	a0005	1
6	a0006	1
7	a0007	0
8	a0008	1
9	a0009	0
10	a0010	1
11	a0011	0
12	a0012	0

Gambar 3.5 Daftar Sinyal Suara Jantung

Data sinyal jantung PCG diambil dari *Physionet Cardiology Challenge* 2016. Data yang digunakan untuk pelatihan sebanyak 3240 data sinyal suara PCG normal dan tidak normal, dan data yang digunakan untuk diprediksi untuk mengetahui kesesuaian pengujian pada klasifikasi sebanyak 301 data.

Keseluruhan sinyal suara jantung memiliki durasi dari 5 detik sampai 120 detik. Nama *file* sinyal suara jantung disimpan kedalam tabel dan diberi tanda angka 0 untuk sinyal normal dan angka 1 untuk sinyal abnormal.

3. Normalisasi

```
def loadData(trainingFolder):
    wav = pd.read_csv('Pictures/LSTM/{}/REFERENCE.csv'.format(trainingFolder), header=None, names =['filename', 'outcome'])
    wav_list = []
    wav_name = []

    wav.outcome.replace(to_replace=-1, value=0, inplace=True)

    for fname in wav.filename:
        path = "Pictures/LSTM/{}/{}.wav".format(trainingFolder, fname)
        sampFreq, snd = wavfile.read(path)
        snd = snd/(snd(max))
        wav_list.append(snd)
        wav_name.append(fname)
    return wav_list, wav
```

Gambar 3.6 Program normalisasi

Pada tahap ini semua data sinyal suara PCG akan dinormalisasi. Tujuan dari normalisasi ini agar amplitudo setiap sinyal suara jantung PCG maksimal sama dengan satu.

4. Segmentasi

```
def shortenTo9Sec(sndList):
    snd9SecList = []
    for snd in sndList:
        sndNP = np.asarray(snd)
        if sndNP.size <= 16000:
            snd9sec = sndNP
        elif sndNP.argmax() > 8000 and (sndNP.size - sndNP.argmax()) >= 8000:
            snd9sec = snd[sndNP.argmax() - 8000:sndNP.argmax()+8000]
        elif sndNP.argmax() < 8000 and sndNP.size - sndNP.argmax() > 8000:
            snd9sec = sndNP[0:16000]
        elif sndNP.argmax() > 8000 and sndNP.size - sndNP.argmax() <= 8000:
            snd9sec = sndNP[sndNP.size - 16000:sndNP.size]
        snd9SecList.append(snd9sec)
    return snd9SecList
```

Gambar 3.7 Program segmentasi

Segmentasi sinyal suara jantung dilakukan dengan membagi sinyal suara jantung ke dalam 16.000 data. Dengan segmentasi 16.000 data diharapkan akan terdapat jumlah siklus sinyal suara jantung yang cukup untuk proses analisis berikutnya. Selain itu, jumlah tersebut dianggap tidak terlalu banyak agar proses pelatihan tidak memakan daya di komputer terlalu lama.

5. *Fast Fourier Transform*

```
def fftProcess4(sndList, cutOffidx, offset):
    sndFFTList = []
    fList = []
    NFFT = 16384
    Fs = 2000
    for idx, snd in enumerate(sndList):
        L = len(snd)
        Ypre = fft(snd, NFFT)/L
        Y = 2*np.abs(Ypre[0:NFFT//2])
        sndFFTList.append(average(Y[0:cutOffidx],
        f = Fs/2*np.linspace(0.0, 1, NFFT//2+1)
        fList.append(f)
    return sndFFTList, fList
```

Gambar 3.8 Program *fast fourier transform*

Fast fourier transform merupakah salah satu algoritma yang merubah sinyal yang sebelumnya berada pada *domain* waktu menjadi *domain* frekuensi sehingga diciptakan spektrum yang menyusun komponen frekuensi. Lewat spektrum tersebut nampak terlihat bentuk dari bentuk sinyal suara jantung normal dan sinyal suara jantung tidak normal. NFFT adalah jumlah dari titik FFT. Perhitungan FFT akan lebih efisien ketika jumlah sampel dari FFT jika dapat difaktorkan dengan bilangan prima terkecil. Setiap rata – rata dari titik ke empat akan disimpan ke dalam variabel sndFFTList.

6. Proses Pelatihan LSTM

Proses pelatihan pada jaringan LSTM diilustrasikan pada Gambar L3.1, dilakukan dengan menjadikan tiap siklus sinyal menjadi sebuah model generatif. Setiap satu siklus sinyal PCG dilatih, dan tiap siklus menjadi fitur dalam dataset pelatihan. Satu siklus sinyal yang dilatihkan juga disertakan data latih label. Sinyal PCG satu siklus berpasangan dengan label membentuk sebuah data yang utuh kemudian menjadi data untuk diingat saat proses training menggunakan LSTM.

Proses pelatihan LSTM dilakukan dengan menggunakan 4 variasi banyaknya *layer*. Variasi dari jumlah *hidden layer* tersebut dapat dilihat pada Tabel 3.1. Menggunakan *batch size* 500 berarti program akan mengirimkan sebanyak 500 data ke *neural network* saat training, dan 500 iterasi berarti program akan mengulang pelatihan sebanyak 500 kali.

Tabel 3.1 Variasi *layer*

No	Input	*Hidden layer* 1	*Hidden layer* 2	*Hidden layer* 3	*Hidden layer* 4	*Hidden layer* 5
1	615	1024	2048	2048	512	512
2	615	1024	2048	2048	1024	512
3	615	1024	2048	2048	1024	1024
4	615	1024	2048	2048	2048	1024

Dengan 5 *hidden layer* pada LSTM yang jumlahnya bervariasi dari 512 hingga 1024 *layer* diharapkan pelatihan model dapat menghasilkan skor akurasi yang tinggi.

7. Proses Pelatihan BiLSTM

Dalam pengerjaan pelatihan Bidirectional LSTM, input yang dimasukan ke dalam BiLSTM ada 2 variasi yakni input *forward* dan input *backward*. Output dari lapisan arsitektur ini digabungkan menjadi satu. Dengan menerapkan arsitektur *layer* ini, model bisa mempelajari data masa lalu dan data masa mendatang untuk setiap sekuen input.

Proses pelatihan ini dilakukan dengan menggunakan 4 variasi banyaknya *layer*. Variasi dari jumlah *hidden layer* tersebut dapat dilihat pada Tabel 3.2. Menggunakan *batch size* 500 berarti program akan mengirimkan sebanyak 500 data ke *neural network* saat training, dan 500 iterasi berarti program akan mengulang pelatihan sebanyak 500 kali.

Tabel 3.2 Variasi *layer* pelatihan BiLSTM

No	Input	*Hidden layer* 1	*Hidden layer* 2	*Hidden layer* 3	*Hidden layer* 4	*Hidden layer* 5
1	615	512	256	128	64	32
2	615	256	128	64	32	16
3	615	128	64	32	16	8
4	615	64	32	16	8	8

Dengan 5 *hidden layer* pada BiLSTM yang jumlahnya bervariasi dari 8 hingga 512 *layer* diharapkan ada empat alternatif hasil sehingga dapat dipilih alternatif mana yang menghasilkan skor akurasi paling tinggi.

8. Keluaran Hasil LSTM dan BiLSTM

Setelah LSTM dan BiLSTM sudah dilatih akan menyimpan bobot, arstitektur, detail akurasi dan *loss,* dan *optimizer state.* Ini berarti dapat memuat dan menggunakan model secara langsung, tanpa harus melakukan pelatihan ulang untuk mengklasifikasikan data. Tahap klasifikasi hanya bisa dijalankan langsung setelah tahap pelatihan selesai, sehingga hasil pelatihan langsung digunakan secara internal oleh tahap klasifikasi.

3.2 Analisis Parameter Uji

Pada penelitian ini digunakan parameter nilai akurasi, sensitifitas, *specificity*, *positive predictive* value, dan nilai F1. Hal ini karena pada klasifikasi sinyal suara jantung PCG dengan menggunakan metode LSTM dan BiLSTM menghasil nilai keluaran tersebut. Hasil akurasi yang diharapkan dari masing masing metode adalah diatas 80% karena dengan nilai tersebut proses klasifikasi dianggap akurat.

Tabel 3.3 Confusion Matrix

	Relevant	***Not-Relevant***
Retrieved	*True Positive (TP)*	*False Positive (FP)*
Not-Retrieved	*False Negative (FN)*	*True Negative (TN)*

Pada tabel 3.3 *True Positive* (TP) adalah nilai hasil klasifikasi positif dan nilai yang sebenarnya sama positif. *False Positive* (FP) adalah nilai hasil klasifikasi positif dan nilai yang sebenarnya negatif. *True Negative* (TN) adalah nilai hasil klasifikasi negatif dan nilai yang sebenarnya sama negatif. *False Negative* (FN) adalah nilai hasil klasifikasi negatif dan nilai yang sebenarnya positif. Untuk menentukan *accuracy, precision, specificity,* dan *f1-score*, dibutuhkan nilai TP, TN, FP dan FN.

BAB IV
HASIL DAN PEMBAHASAN

Dalam bab ini penulis akan menerangkan sebagian hasil pengujian dari hasil penelitian tugas akhir ini.

4.1 Pengujian Normalisasi

Pengujian normalisasi ini dengan menggunakan aplikasi Google Colaboratory berbasis Phyton.

4.1.1 Tujuan Normalisasi

Pengujian ini dikerjakan untuk membuat nilai amplitudo sinyal suara jantung paling tinggi sama dengan satu.

4.1.2 Alat yang digunakan pada Pengujian Normalisasi

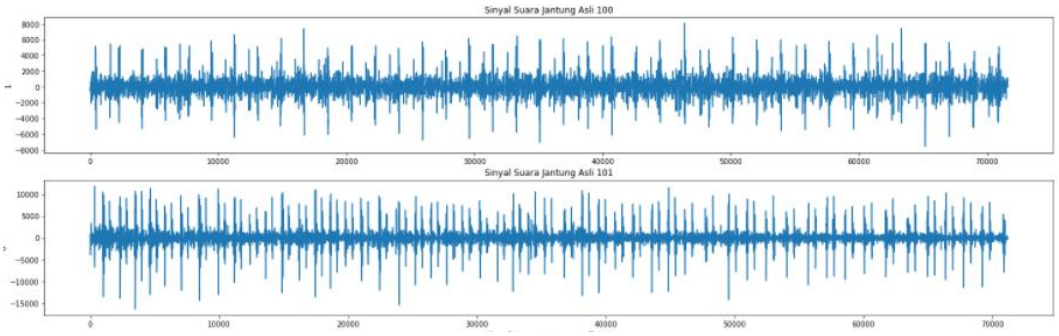
Untuk mengerjakan percobaan ini maka dibutuhkan beberapa alat sebagai berikut.

1. Komputer / Laptop
2. Google Colaboratory

4.1.3 Pro sedur Pengujian Normalisasi

1. Membuka Google Colaboratory.
2. Jalankan blok program normalisasi.

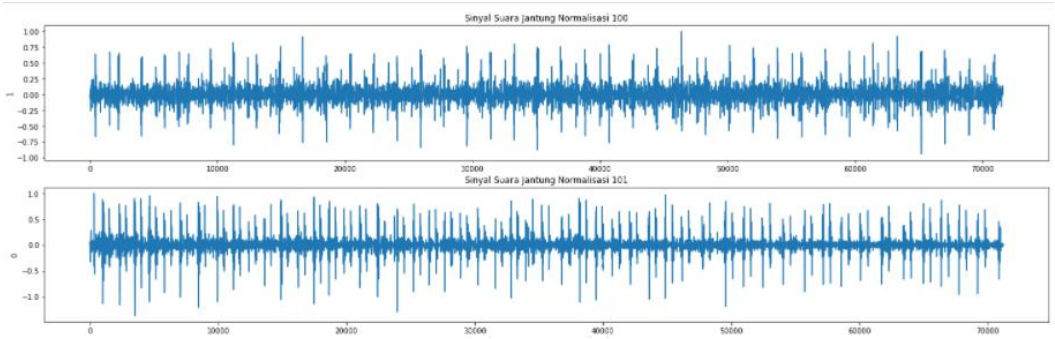
4.1.4 Hasil Pengujian Normalisasi



Gambar 4.1 Sinyal sebelum normalisasi

Pada Gambar 4.1 adalah gambar sinyal suara jantung *phococardiogram* sebelum dilakukan normalisasi. Terdapat 2 grafik yang divisualisasikan, yang

pertama data sinyal suara jantung 100 dan data sinyal suara jantung 101. Pada sinyal suara jantung 100 memiliki *range* amplitudo paling rendah -6.000 sampai amplitudo paling tinggi 8.000. Pada sinyal suara jantung 101 memili *range* amplitudo paling rendah -15.000 sampai amplitudo paling tinggi 10.000.



Gambar 4.2 Sinyal Setelah Normalisasi

Pada Gambar 4.2 merupakan grafik hasil keluaran setelah adanya proses normalisasi. Sehingga pada contoh kedua sinyal suara jantung 100 dan 101 menjadi memiliki *range* amplitude paling rendah -1.00 sampai amplitudo paling tinggi 1.00.

4.2 Pengujian Segmentasi

Pengujian Segmentasi adalah untuk menyesuaikan data input sinyal suara jantung sesuai dengan yang dibutuhkan saat proses training.

4.2.1 Tujuan Pengujian Segmentasi

Tujuan dari proses ini adalah memotong data sinyal suara jantung sebanyak 16.000 data. Dengan dipotong menjadi 16.000 data agar program bisa menganalisis sinyal suara jantung normal dan abnormal serta komputer tidak membutuhkan waktu yang sangat lama saat memproses.

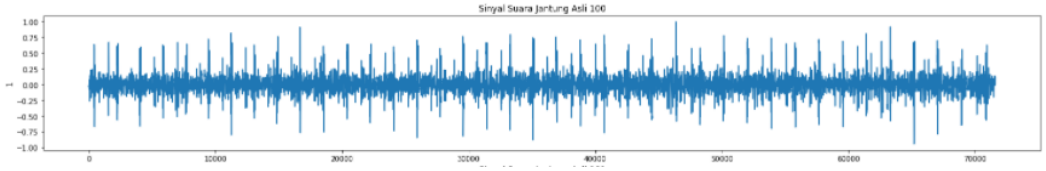
4.2.2 Alat yang digunakan untuk Pengujian Segmentasi

1. Komputer / laptop
2. Google Colaboratory

4.2.3 Prosedur Pengujian Segmentasi

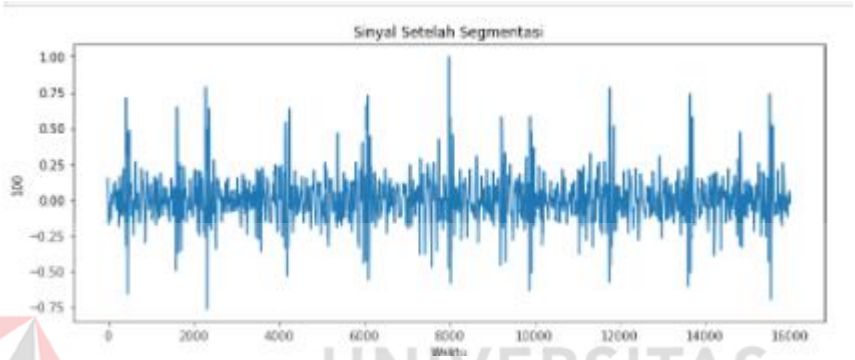
1. Membuka Google Colaboratory.
2. Jalankan blok program segmentasi.

4.2.4 Hasil Pengujian Segmentasi



Gambar 4.3 Sinyal sebelum segmentasi

Pada Gambar 4.3 sinyal suara jantung *phococardiogram* dari Physionet sebelumnya memiliki sampai 70.000 data.



Gambar 4.4 Sinyal setelah segmentasi

Pada Gambar 4.4 setelah proses segmentasi sinyal suara jantung *phonocardiogram* yang sebelumnya memiliki sampai 70.000 data dipotong menjadi 16.000 data.

4.3 Pengujian *Fast Fourier Transform*

Pengujian *Fast Fourier Transform* ini untuk data input LSTM dan BiLSTM.

4.3.1 Tujuan Fast Fourier Transform

Tujuan dari pengerjaan proses *fast fourier transfrom* ini yaitu untuk mengubah sinyal suara jantung dalam domain waktu menjadi sinyal dalam domain frekuensi, artinya proses sinyal suara jantung menjadi dalam bentuk digital berupa gelombang spectrum suara yang berbasis frekuensi.

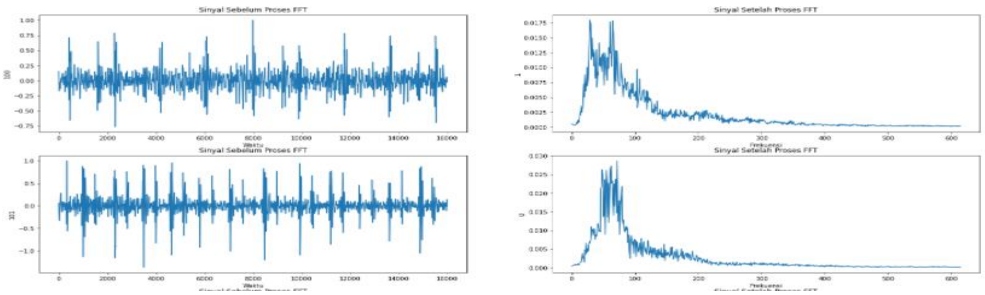
4.3.2 Alat yang digunakan pada Pengujian *Fast Fourier Transform*

1. Komputer / laptop.
2. Google Colaboratory

4.3.3 Prosedur Pengujian Fast Fourier Transform

1. Membuka Google Colaboratory
2. Jalankan blok program *fast fourier transform*.

4.3.4 Hasil Pengujian Fast Fourier Transform



Gambar 4.5 Hasil pengujian FFT

Dari Gambar 4.5 digambarkan sinyal suara jantung ditransformasikan ke dalam domain frekuensi.

4.4 Pengujian Proses Pelatihan LSTM

Pengerjaan proses pelatihan dijalankan dengan menggunakan aplikasi Google Colaboratory berbasis Python.

4.4.1 Tujuan Proses Pelatihan LSTM

Pengujian ini dikerjakan untuk membuat pengerjaan pelatihan LSTM bisa digunakan dan berfungsi dengan baik, serta mendapatkan skor ketepatan yang tinggi.

4.4.2 Alat yang digunakan pada Pengujian Proses Pelatihan LSTM

1. Komputer/Laptop
2. Google Colaboratory

4.4.3 Prosedur Pengujian Proses Pelatihan LSTM

Prosedur pengujian :

1. Buka Google Colaboratory.
2. Jalankan blok program pelatihan LSTM.

UNIVERSITAS Dinamika

4.4.4 Hasil Pengujian Proses Pelatihan LSTM

Supaya dapat mengetahui tingkat keberhasilan dari pengerjaan pelatihan ini diperhatikan dari nilai akurasinya. Berikut hasil dari pengujian yang telah dikerjakan.

1. Hasil Pengujian Kesatu

```
[ ]  model = Sequential()
     layers = {'input': 615, 'hidden1':1024, 'hidden2': 2048, 'hidden3': 2048, 'hidden4': 512, 'hidden5': 512, 'output': 1}
```

Gambar 4.6 Desain layer LSTM pengujian kesatu

Pada Gambar 4.6 program pengujian pertama menggunakan input sejumlah 615, 1024 *hidden layer* pertama, 2048 *hidden layer* kedua, 2048 *hidden layer* ketiga, 512 *hidden layer* keempat, dan terakhir 512 *hidden layer* kelima.

Pada Gambar L4.1 merupakan hasil pengujian kesatu dimana pengujian kesatu ini terdapat dua grafik. Grafik pertama menggambarkan dua sumbu loss dan *epoch*. Grafik kedua menggambarkan dua sumbu akurasi dan *epoch*. dua variabel yang digambarkan adalah variabel train dan variabel *test*. Variabel train digambarkan dengan warna biru dan variabel *test* digambarkan dengan warna orange. Pada pengujian kesatu sampai *epoch* 500 hasil skor akurasi training adalah 0.8849, akurasi validasi 0.8909.

Dari model yang sudah ditraining pada pengujian pertama, program melakukan prediksi dari data *test*. Pada Gambar L6.1 ditampilkan hasil data *test* yang telah diprediksi 117 data positif dengan nilai yang sesungguhnya positif, 121 data negative dengan nilai yang sesungguhnya negative, 29 data positif dengan nilai yang sesungguhnya negative, 34 data negative dengan nilai yang sesungguhnya positif.

2. Hasil Pengujian Kedua

```
[ ]  model = Sequential()
     layers = {'input': 615, 'hidden1':1024, 'hidden2': 2048, 'hidden3': 2048, 'hidden4': 1024, 'hidden5': 512, 'output': 1}
```

Gambar 4.7 Desain layer pengujian LSTM kedua

Pada Gambar 4.7 program pengujian kedua menggunakan input sejumlah 615, 1024 *hidden layer* pertama, 2048 *hidden layer* kedua, 2048 *hidden layer* ketiga, 1024 *hidden layer* keempat, dan terakhir 512 *hidden layer* kelima.

Pada Gambar L4.2 merupakan hasil pengujian kedua dimana pengujian kedua ini terdapat dua grafik. Grafik pertama menggambarkan dua sumbu loss dan *epoch*. Grafik kedua menggambarkan dua sumbu akurasi dan *epoch*. Dua variabel yang digambarkan adalah variabel train dan variabel *test*. Variabel train digambarkan dengan warna biru dan variabel *test* digambarkan dengan warna orange. Pada pengujian kedua sampai *epoch* 500 hasil skor akurasi training adalah 0.8973, akurasi validasi 0.9280.

Dari model yang sudah ditraining pada pengujian kedua, program melakukan prediksi dari data *test*. Pada Gambar L6.2 ditampilkan hasil data *test* yang telah diprediksi 87 data positif dengan nilai yang sesungguhnya positif, 135 data negative dengan nilai yang sesungguhnya negative, 15 data positif dengan nilai yang sesungguhnya negative, 64 data negative dengan nilai yang sesungguhnya positif.

3. Hasil Pengujian Ke 3

```
[ ]  model = Sequential()
     layers = {'input': 615, 'hidden1':1024, 'hidden2': 2048, 'hidden3': 2048, 'hidden4': 1024, 'hidden5': 1024, 'output': 1}
```

Gambar 4.8 Desain layer LSTM pengujian ketiga

Pada Gambar 4.8 program pengujian ketiga menggunakan input sejumlah 615, 1024 *hidden layer* pertama, 2048 *hidden layer* kedua, 2048 *hidden layer* ketiga, 1024 *hidden layer* keempat, dan terakhir 1024 *hidden layer* kelima.

Pada Gambar L4.3 merupakan hasil pengujian ketiga dimana pengujian ketiga ini terdapat dua grafik. Grafik pertama menggambarkan dua sumbu loss dan *epoch*. Grafik kedua menggambarkan dua sumbu akurasi dan *epoch*. Dua variabel yang digambarkan adalah variabel train dan variabel *test*. Variabel train digambarkan dengan warna biru dan variabel *test* digambarkan dengan warna orange. Pada pengujian ketiga sampai *epoch* 500 hasil skor akurasi training adalah 0.9101, akurasi validasi 0.8909.

Dari model yang sudah ditraining pada pengujian ketiga, program melakukan prediksi dari data *test*. Pada Gambar L6.3 ditampilkan hasil data *test* yang telah diprediksi 121 data positif dengan nilai yang sesungguhnya positif, 125 data negative dengan nilai yang sesungguhnya negative, 25 data positif dengan nilai

yang sesungguhnya negative, 30 data negative dengan nilai yang sesungguhnya positif.

4. Hasil Pengujian Keempat

```
[ ]  model = Sequential()
     layers = {'input': 615, 'hidden1':1024, 'hidden2': 2048, 'hidden3': 2048, 'hidden4': 2048, 'hidden5': 1024, 'output': 1}
```

Gambar 4.9 Desain layer pengujian LSTM keempat

Pada Gambar 4.9 program pengujian keempat menggunakan input sejumlah 615, 1024 *hidden layer* pertama, 2048 *hidden layer* kedua, 2048 *hidden layer* ketiga, 2048 *hidden layer* keempat, dan terakhir 1024 *hidden layer* kelima.

Pada Gambar L4.4 merupakan hasil pengujian keempat dimana pengujian kempat ini terdapat dua grafik. Grafik pertama menggambarkan dua sumbu loss dan *epoch*. Grafik kedua menggambarkan dua sumbu akurasi dan *epoch*. dua variabel yang digambarkan adalah variabel train dan variabel *test*. Variabel train digambarkan dengan warna biru dan variabel *test* digambarkan dengan warna orange. Pada pengujian keempat sampai *epoch* 500 hasil skor akurasi training adalah 0.8951, akurasi validasi 0.9270.

Dari model yang sudah ditraining pada pengujian keempat, program melakukan prediksi dari data *test*. Pada Gambar L6.4 ditampilkan hasil data *test* yang telah diprediksi 83 data positif dengan nilai yang sesungguhnya positif, 138 data negative dengan nilai yang sesungguhnya negative, 12 data positif dengan nilai yang sesungguhnya negative, 68 data negative dengan nilai yang sesungguhnya positif.

Pada Tabel L8.1 dan Tabel L8.2 disimpulkan hasil skor akurasi *training* dan akurasi validasi dari keempat pengujian yang setiap pengujian menggunakan jumlah *layer* yang berbeda.

4.5 Pengujian Proses Pelatihan Bidirectional LSTM

Pengujian pengerjaan pelatihan dikerjakan dengan menggunakan aplikasi Google Colaboratory berbasis Python.

4.5.1 Tujuan Proses Pelatihan Bidirectional LSTM

Pengujian ini dilakukan untuk membuat pengerjaan pelatihan BiLSTM bisa digunakan dan berfungsi dengan baik, serta mendapatkan skor ketepatan yang tinggi.

4.5.2 Alat yang digunakan pada Pengujian Proses Pelatihan Bidirectional LSTM

1. Komputer/Laptop
2. Google Colaboratory

4.5.3 Prosedur Pengujian Proses Pelatihan Bidirectional LSTM

Prosedur pengujian :

1. Buka Google Colaboratory.
2. Semua data sinyal suara jantung harus dilakukan proses normalisasi, *fast fourier transform*, dan segmentasi.

4.5.4 Hasil Pengujian Proses Pelatihan Bidirectional LSTM

Supaya dapat mengetahui tingkat keberhasilan dari pengerjaan pelatihan *Bidirectional* LSTM ini diperhatikan dari nilai akurasinya. Berikut hasil dari pengujian yang telah dikerjakan.

1. Hasil Pengujian Pertama

```
model = Sequential()
layers = {'input': 615, 'hidden1':512, 'hidden2': 256, 'hidden3': 128,
          'hidden4': 64, 'hidden5': 32, 'output': 1}

model.add(Bidirectional(LSTM(layers['input'], return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden1'],return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden2'],return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden3'],return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden4'],return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden5'],return_sequences=False)))
model.add(Dropout(0.2))
model.add(Dense(layers['output']))
model.add(Activation("sigmoid"))
model.compile(loss="binary_crossentropy", optimizer="adam", metrics=['accuracy'])
```

Gambar 4.10 Desain layer BiLSTM pengujian pertama

Pada Gambar 4.10 program pengujian BiLSTM pertama menggunakan input sejumlah 615, 512 *hidden layer* pertama, 256 *hidden layer* kedua, 128 *hidden layer* ketiga, 64 *hidden layer* keempat, dan terakhir 32 *hidden layer* kelima.

Pada gambar L5.1 merupakan hasil pengujian kesatu dimana pengujian kesatu ini terdapat dua grafik. Grafik pertama menggambarkan dua sumbu loss dan *epoch*. Grafik kedua menggambarkan dua sumbu akurasi dan *epoch.* dua variabel yang digambarkan adalah variabel train dan variabel *test.* Variabel train digambarkan dengan warna biru dan variabel *test* digambarkan dengan warna orange. Pada pengujian kesatu sampai *epoch* 500 hasil skor akurasi training adalah 0.9303, akurasi validasi 0.8961.

Dari model yang sudah ditraining pada pengujian kesatu, program melakukan prediksi dari data *test.* Pada Gambar L7.1 ditampilkan hasil data *test* yang telah diprediksi 110 data positif dengan nilai yang sesungguhnya positif, 135 data negative dengan nilai yang sesungguhnya negative, 15 data positif dengan nilai yang sesungguhnya negative, 41 data negative dengan nilai yang sesungguhnya positif.

2. Hasil Pengujian Kedua

```
model = Sequential()
layers = {'input': 615, 'hidden1':256, 'hidden2': 128, 'hidden3': 64,
          'hidden4': 32, 'hidden5': 16, 'output': 1}

model.add(Bidirectional(LSTM(layers['input'], return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden1'],return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden2'],return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden3'],return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden4'],return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden5'],return_sequences=False)))
model.add(Dropout(0.2))
model.add(Dense(layers['output']))
model.add(Activation("sigmoid"))
model.compile(loss="binary_crossentropy", optimizer="adam", metrics=['accuracy'])
```

Gambar 4.11 Desain layer pengujian BiLSTM kedua

Pada Gambar 4.11 program pengujian pertama menggunakan input sejumlah 615, 256 *hidden layer* pertama, 128 *hidden layer* kedua, 64 *hidden layer* ketiga, 32 *hidden layer* keempat, dan terakhir 16 *hidden layer* kelima.

Pada Gambar L5.2 merupakan hasil pengujian kedua dimana pengujian kedua ini terdapat dua grafik. Grafik pertama menggambarkan dua sumbu loss dan *epoch*. Grafik kedua menggambarkan dua sumbu akurasi dan *epoch*. Dua variabel yang digambarkan adalah variabel train dan variabel *test.* Variabel train digambarkan dengan warna biru dan variabel *test* digambarkan dengan warna orange. Pada

pengujian kedua sampai *epoch* 500 hasil skor akurasi training adalah 0.9352, akurasi validasi 0.8776.

Dari model yang sudah ditraining pada pengujian kedua, program melakukan prediksi dari data *test*. Pada Gambar L7.2 ditampilkan hasil data *test* yang telah diprediksi 117 data positif dengan nilai yang sesungguhnya positif, 135 data negative dengan nilai yang sesungguhnya negative, 15 data positif dengan nilai yang sesungguhnya negative, 34 data negative dengan nilai yang sesungguhnya positif.

3. Hasil Pengujian Ketiga

```
model = Sequential()
layers = {'input': 615, 'hidden1':128, 'hidden2': 64, 'hidden3': 32,
          'hidden4': 16, 'hidden5': 8, 'output': 1}

model.add(Bidirectional(LSTM(layers['input'], return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden1'],return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden2'],return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden3'],return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden4'],return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden5'],return_sequences=False)))
model.add(Dropout(0.2))
model.add(Dense(layers['output']))
model.add(Activation("sigmoid"))
model.compile(loss="binary_crossentropy", optimizer="adam", metrics=['accuracy'])
```

Gambar 4.12 Desain layer BiLSTM pengujian ketiga

Pada Gambar 4.12 program pengujian ketiga menggunakan input sejumlah 615, 128 *hidden layer* pertama, 64 *hidden layer* kedua, 32 *hidden layer* ketiga, 16 *hidden layer* keempat, dan terakhir 8 *hidden layer* kelima.

Pada Gambar L5.3 merupakan hasil pengujian ketiga dimana pengujian ketiga ini terdapat dua grafik. Grafik pertama menggambarkan dua sumbu loss dan *epoch*. Grafik kedua menggambarkan dua sumbu akurasi dan *epoch*. dua variabel yang digambarkan adalah variabel train dan variabel *test*. Variabel train digambarkan dengan warna biru dan variabel *test* digambarkan dengan warna orange. Pada pengujian ketiga sampai *epoch* 500 hasil skor akurasi training adalah 0.9541, akurasi validasi 0.8899.

Dari model yang sudah ditraining pada pengujian ketiga, program melakukan prediksi dari data *test*. Pada Gambar L7.3 ditampilkan hasil data *test* yang telah diprediksi 135 data positif dengan nilai yang sesungguhnya positif, 133 data negative dengan nilai yang sesungguhnya negative, 17 data positif dengan nilai

yang sesungguhnya negative, 16 data negative dengan nilai yang sesungguhnya positif.

4. Hasil Pengujian Ke 4

```
model = Sequential()
layers = {'input': 615, 'hidden1':64, 'hidden2': 32, 'hidden3': 16,
          'hidden4': 8, 'hidden5': 8, 'output': 1}

model.add(Bidirectional(LSTM(layers['input'], return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden1'],return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden2'],return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden3'],return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden4'],return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(layers['hidden5'],return_sequences=False)))
model.add(Dropout(0.2))
model.add(Dense(layers['output']))
model.add(Activation("sigmoid"))
model.compile(loss="binary_crossentropy", optimizer="adam", metrics=['accuracy'])
```

Gambar 4.13 Desain Layer Pengujian BiLSTM 4

Pada Gambar 4.13 program pengujian keempat menggunakan input sejumlah 615, 64 *hidden layer* pertama, 32 *hidden layer* kedua, 16 *hidden layer* ketiga, 8 *hidden layer* keempat, dan terakhir 8 *hidden layer* kelima.

Pada Gambar L5.4 merupakan hasil pengujian keempat dimana pengujian keempat ini terdapat 2 grafik. Grafik pertama menggambarkan 2 sumbu loss dan *epoch*. Grafik kedua menggambarkan 2 sumbu akurasi dan *epoch.* 2 variabel yang digambarkan adalah variabel train dan variabel *test.* Variabel train digambarkan dengan warna biru dan variabel *test* digambarkan dengan warna orange. Pada pengujian keempat sampai *epoch* 500 hasil skor akurasi training adalah 0.9484, akurasi validasi 0.8899.

Dari model yang sudah ditraining pada pengujian keempat, program melakukan prediksi dari data *test.* Pada Gambar L7.4 ditampilkan hasil data *test* yang telah diprediksi 135 data positif dengan nilai yang sesungguhnya positif, 129 data negative dengan nilai yang sesungguhnya negative, 16 data positif dengan nilai yang sesungguhnya negative, 21 data negative dengan nilai yang sesungguhnya positif.

4.6 Analisis Parameter Uji Metode LSTM

Hasil dari klasifikasi yang sudah dilakukan sebanyak empat kali dan menghasilkan tabel *confusion matrix* akan dihitung untuk menganalisis unjuk kerja

metode LSTM ,lima parameter uji antara lain: *Accuracy, Precision, Recall, Specificity*, dan *F1-Score.* Perhitungan parameter uji ini mengacu pada persamaan 2.7, 2.8, 2.9, 2.10.

4.7.1 Pengujian LSTM Kesatu

$$Accuracy = \frac{(117 + 121)}{(117 + 34 + 121 + 29)} = 0.79$$

$$Precision = \frac{117}{(117 + 29)} = 0.80$$

$$Recall = \frac{117}{(117 + 34)} = 0.77$$

$$Specificity = \frac{121}{(121 + 29)} = 0.80$$

$$F1 - Score = 2 \times \frac{0.80 \times 0.80}{0.80 + 0.80} = 0.78$$

4.7.2 Pengujian LSTM Kedua

$$Accuracy = \frac{(87 + 135)}{(87 + 64 + 135 + 15)} = 0.74$$

$$Precision = \frac{87}{(87 + 15)} = 0.83$$

$$Recall = \frac{87}{(87 + 64)} = 0.58$$

$$Specificity = \frac{135}{(135 + 15)} = 0.90$$

$$F1 - Score = 2 \times \frac{0.83 \times 0.58}{0.83 + 0.58} = 0.69$$

UNIVERSITAS Dinamika

4.7.3 Pengujian LSTM Ketiga

$$Accuracy = \frac{(121 + 125)}{(121 + 30 + 125 + 25)} = 0.81$$

$$Precision = \frac{121}{(121 + 25)} = 0.83$$

$$Recall = \frac{121}{(121 + 30)} = 0.80$$

$$Specificity = \frac{125}{(125 + 25)} = 0.83$$

$$F1 - Score = 2 \times \frac{0.83 \times 0.80}{0.83 + 0.80} = 0.81$$

4.7.4 Pengujian LSTM Keempat

$$Accuracy = \frac{(83 + 138)}{(83 + 68 + 138 + 12)} = 0.73$$

$$Precision = \frac{83}{(83 + 12)} = 0.87$$

$$Recall = \frac{83}{(83 + 68)} = 0.55$$

$$Specificity = \frac{138}{(138 + 12)} = 0.92$$

$$F1 - Score = 2 \times \frac{0.87 \times 0.55}{0.87 + 0.55} = 0.67$$

4.7 Analisis Parameter Uji BiLSTM

Hasil dari klasifikasi yang sudah dilakukan sebanyak empat kali dan menghasilkan tabel *confusion matrix* akan dihitung untuk menganalisis unjuk kerja

metode BiLSTM ,lima parameter uji antara lain: *Accuracy*, *Precision*, *Recall*, *Specificity*, dan *F1-Score*. Perhitungan parameter uji ini mengacu pada persamaan 2.7, 2.8, 2.9, 2.10.

4.7.1 Pengujian BiLSTM Kesatu

$$Accuracy = \frac{(135 + 110)}{(135 + 110 + 41 + 15)} = 0.81$$

$$Precision = \frac{110}{(110 + 15)} = 0.83$$

$$Recall = \frac{110}{(110 + 41)} = 0.87$$

$$Specificity = \frac{135}{(135 + 55)} = 0.86$$

$$F1 - Score = 2 \times \frac{0.83 \times 0.87}{0.83 + 0.87} = 0.85$$

4.7.2 Pengujian BiLSTM Kedua

$$Accuracy = \frac{(117 + 135)}{(117 + 135 + 34 + 15)} = 0.83$$

$$Precision = \frac{117}{(117 + 15)} = 0.88$$

$$Recall = \frac{117}{(117 + 34)} = 0.77$$

$$Specificity = \frac{135}{(135 + 15)} = 0.90$$

$$F1 - Score = 2 \times \frac{0.88 \times 0.77}{0.88 + 0.77} = 0.82$$

4.7.3 Pengujian BiLSTM Ketiga

$$Accuracy = \frac{(135 + 133)}{(135 + 133 + 17 + 16)} = 0.89$$

UNIVERSITAS Dinamika

$$Precision = \frac{135}{(135 + 17)} = 0.88$$

$$Recall = \frac{135}{(135 + 16)} = 0.89$$

$$Specificity = \frac{133}{(133 + 17)} = 0.88$$

$$F1 - Score = 2 \times \frac{0.88 \times 0.89}{0.88 + 0.89} = 0.89$$

4.7.4 Pengujian BiLSTM Keempat

$$Accuracy = \frac{(135 + 129)}{(135 + 129 + 16 + 21)} = 0.87$$

$$Precision = \frac{135}{(135 + 21)} = 0.86$$

$$Recall = \frac{135}{(135 + 16)} = 0.89$$

$$Specificity = \frac{129}{(129 + 21)} = 0.86$$

$$F1 - Score = 2 \times \frac{0.86 \times 0.89}{0.86 + 0.89} = 0.87$$

4.8 Analisis Perbandingan Metode LSTM dan BiLSTM

Tabel 4IV.1 Hasil Prediksi LSTM

Pengujian LSTM	*Test* Accuracy	Precision	Recall	Specificity	F1-Score
1	0.79	0.80	0.77	0.80	0.78
2	0.74	0.83	0.58	0.90	0.69
3	0.81	0.83	0.80	0.83	0.81
4	0.73	0.87	0.55	0.92	0.67

Tabel 4.2 Hasil Prediksi BiLSTM

Pengujian BiLSTM	*Test Accuracy*	*Precision*	*Recall*	*Specificity*	*F1-Score*
1	0.81	0.83	0.87	0.86	0.85
2	0.83	0.88	0.77	0.90	0.82
3	0.89	0.88	0.89	0.88	0.89
4	0.87	0.86	0.89	0.86	0.87

Pada Tabel 4.1 dan Tabel 4.2 merupakan perhitungan dari hasil prediksi metode LSTM dan BiLSTM. *Test Accuracy* adalah rasio prediksi benar (data normal dan data abnormal) dengan jumlah keseluruhan data. *Precision* merupakan rasio prediksi benar data normal dibandingkan dengan jumlah data yang diprediksi normal. *Recall* atau sensitifitas merupakan rasio prediksi benar data normal dibandingkan dengan jumlah data yang benar normal. *Specificity* merupakan kebenaran memprediksi data abnormal dibandingkan dengan jumlah data abnormal. *F1-Score* merupakan perbandingan dari rata-rata *precision* dan *recall*.

Dari perbandingan metode LSTM dan BiLSTM yang sudah dilakukan pengujian dapat disimpulkan bahwa pengujian dengan metode BiLSTM lebih akurat untuk melakukan klasifikasi sinyal suara jantung *phonocardiogram*. Karena BiLSTM merupakan gabungan dari dua LSTM yaitu *input forward* dan *input backward* jadi saat memproses data masukan sinyal diproses dari depan ke belakang dan dari belakang ke depan.

BAB V
PENUTUP

5.1. Kesimpulan

Berdasarkan semua hasil dari pengujian klasifikasi sinyal suara jantung *phonocardiogram* menggunakan metode LSTM dan BiLSTM, maka dapat diambil kesimpulan sebagai berikut.

1. Bedarsarkan hasil pengujian sebanyak empat kali untuk metode LSTM didapatkan rata – rata *precision* 83%, rata – rata *recall* 67%, rata – rata *specificity* 86%, dan *F1-Score* 73%. Sedangkan untuk metode BiLSTM didapatkan rata – rata *precision* 86%, rata – rata *recall* 85%, rata – rata *specificity* 87%, dan *F1-Score* 85%.
2. Hasil *test accuracy* tertinggi pada setiap pengujian untuk metode LSTM mendapatkan nilai tertinggi sebesar 81% dan untuk metode BiLSTM mendapatkan nilai tertinggi sebesar 89%.

5.2. Saran

Dari kesimpulan yang sudah dibuat, maka ada beberapa saran untuk melakukan pengembangan pada penelitian ini agar lebih baik.

1. Menambahkan preproses seperti *denoising* untuk membuang *noise* pada sinyal suara jantung *phonocardiogram*.
2. Mengubah perancangan model, *optimizer* dan variasi *layer* yang digunakan.
3. Menambah dataset sinyal suara jantung *phonocardiogram*.

UNIVERSITAS Dinamika

DAFTAR PUSTAKA

Behbani, S. (2019). A hybrid Algortm fo Heart Sound Segmentation based on Phonocardioram. *Journal of Medical Engineering & Technology*, 1-15.

Chung, J., Gulcehre, C., Cho, K., Bengio, & Y. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *Emprint Ervix*. Retrieved from http://arxiv .org/abs/1412.3555v 1

Debbal, S. M. (2008). Computerized Heart Sounds Analysis. *Computer in Biology and Medicine, 38*(2), 263-280.

Gharehbaghi, A. B. (2015). A Novel Method for Discrimination between Innocent and Pathological Heart Murmurs. *Medical Engineering & Physics, 37*(7), 674-682.

Graves, A. (2013). Generating Sequences with Recurrent Neural Networks. *Computation Science*.

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation, 9*, 1735-1780.

Ismail, S., Siddiq, I., & Akram , U. (2018). Localization dan Classification of Heart Beat in Phonocardiography Signal - a Comprehensive Review. *EURASIP Journal anda Advances Signal, 26*, 1-27.

Junita, V., & Bachtiar, F. A. (2019). Klasifikasi Aktivitas Manusia menggunakan Algoritme Decision Tree C4.5 dan Information Gain untuk Seleksi Fitur. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 3*(10), 9426-9433.

Langley, P. &. (2016). Abnormal heart sounds detected from short duration. *Computing in Cardiology Conference (CinC)*, (pp. 545-548).

Li, S., Li, F., Tang, S., & Xiong, W. (2020). A Review of Computer-Aided Heart Sound Detection Technique. *BiomED Research International, 2020*. doi:https://doi.org/10.1155/2020/5846191

UNIVERSITAS Dinamika

M, T., T, K., G, M., M, Heinzmann, & T, W. (2016). Heart sound classification using deep structured. *Computing in Cardiology Conference (CinC).* Vancouver.

Maisyaroh, S. (2012). *Rancang Bangun Instrumentasi Elektrokardiografi Berbasis Pc Menggunakan Sound Card.* Medan: Fakultas MIPA Jurusan S1 Fisika, Universitas Negeri Medan.

Oktarina, E. S., Puspasari, I., & Jusak, J. (2018). Auskultasu Jarak Jauh untuk Pengukuran dan Perekaman Sinyal Suara Jantung. *Jurnal Rekayasa Elektrika, 14*, 145-220.

Olah, C. (2015, August 27). *Understanding LSTM Networks*. Retrieved from https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Orgaanization, W. H. (n.d.). *Mortality Due Cardiovascular Diseases in World*. Retrieved April 15, 2020, from https://www.medicalnewstoday.com/articles/282929

Sepulveda-Cano, M., Gil, E., & Caste, G. (2011). Selection of Nonstationary Dynamic Features for Obstructive Sleep Apnoea Detection in Children. *EURASIP Journal on Advances in Signal Processing*. doi: https://doi.org/10.1155/2011/538314

Shan, G., Yineng, Z., & Guo, X. (2020). Gated recurrent unit-based heart sound. *Biomedical Engineering Online, 19*, 3-20. doi:https://doi.org/10.1186/s12938-020-0747-x

Sipasulta, R. Y., Lumenta, ST., MT., A. S., & Sompie, ST., MT., S. R. (2014). Simulasi Sistem Pengacak Sinyal Dengan Metode FFT (Fast Fourier Transform). *E-journal Teknik Elektro dan Komputer*(2301-8402).

T, C., S, Y. L., & Ho et al. (2017). S1 and S2 heart sound recognition. *IEEE Transactions on Biomedical, 64*, 372-380.

Taylor, T. (2020, Maret 21). *Heart.* (S. Curreli, Editor) Retrieved April 15, 2020, from Innerbody Research: http://www.innerbody.com/image/card01.html#full-description

UNIVERSITAS Dinamika

W Phanphaisarn, e. a. (2011). Heart detection and diagnosis based on ECG and. *Medical Devices: Evidence and Research*, 133-144.

UNIVERSITAS Dinamika