PERBANDINGAN PENERAPAN ALGORITMA NEURAL NETWORK BACKPROPAGATION DENGAN OPTIMASI ALGORITMA LBFGS DAN SGD UNTUK PREDIKSI PENYAKIT JANTUNG

Guntur Eka Saputra, Karmilasari, Adrian Faisal, dan Ahmad Apandi
Universitas Gunadarma
Jl. Margonda Raya No. 100, Depok, Jawa Barat 16424
{guntur, karmila}@staff.gunadarma.ac.id, adrianfaisal@aol.com, ahmadapandi@staff.gunadarma.ac.id,

ABSTRAK

Penyakit jantung merupakan penyakit tidak menular yang disebabkan kondisi adanya timbunan lemak di pembuluh darah arteri coroner pada jantung yang mengubah peran dan bentuk arteri, serta menghambat aliran darah menuju jantung. Faktor penyakit jantung dapat diprediksi dengan 14 atribut faktor yang dapat mempengaruhi prediksi penyakit jantung. Penelitian ini merupakan penelitian eksperimen dengan menggunakan data sekunder yang diharapkan data tersebut memiliki makna untuk mendapatkan suatu informasi dan pengetahuan. Data mining adalah proses mencari pola atau informasi bermakna dalam data terpilah dengan mengunakan suatu metode atau teknik tertentu. Metode yang digunakan dalam penelitian ini adalah algoritma neural network backpropagation dengan metode optimasi LBFGS dan SGD untuk melihat hasil akurasi perbandingan diantara dua metode optimasi tersebut. Berdasarkan hasil eksperimen ditunjukkan bahwa hasil akurasi dari 14 atribut faktor dengan menggunakan backpropagation dan metode optimasi LBFGS lebih baik dibandingkan SGD. Akurasi dengan metode optimasi LBFGS sebesar 85.3%, sedangkan SGD 83.6%, dan tingkat loss atau error dari LBFGS mendekati 0, yaitu 0.001152, sedangkan SGD sebesar 0.546822. Waktu pemrosesan CPU Times juga memiliki perbedaan yang cukup signifikan, metode LBFGS memproses selama 46.5 ms, sedangkan SGD sebesar 908 ms.

Kata Kunci : *penyakit jantung, data mining, backpropagation, optimasi, LBFGS, SGD*

PENDAHULUAN

Masalah kesehatan merupakan masalah yang terus berkembang. Salah satu masalah kesehatan yang dihadapi adalah meningkatnya penyakit tidak menular, terutama penyakit jantung dan pembuluh darah yang termasuk ke dalam jenis beban penyakit ganda (*double burden of disease*). Penderita penyakit jantung di Indonesia dari hasil riskesdas tahun 2007 sebesar 7,2% per 1000 penduduk dan data WHO pada tahun 2005 yang mengalami serangan jantung sebesar 7,6 juta dari 17,5 juta kematian [1]. Pada tahun 2015, data menunjukkan bahwa 70% kematian di dunia disebabkan oleh Penyakit Tidak Menular (PTM) yaitu 39,5 juta dari 56, juta jiwa kematian. Dari seluruh angka kematian akibat PTM tersebut, 45% disebabkan oleh Penyakit Jantung dan pembuluh darah, yaitu 17,7 juta dari 39,5 juta jiwa kematian. Pada tahun 2018 hasil riskesdas menunjukkan prevalensi Penyakit Jantung berdasarkan diagnosis dokter di Indonesia sebesar 1.5% (prevalensi nasional), dengan peringkat prevalensi tertinggi yaitu Kalimantan Utara (2,2%), DIY (2%), Gorontalo (2%). Penyakit jantung terdiri dari penyakit jantung koroner stabil tanpa gejala, angina pektoris stabil, dan Sindrom Koroner Akut (SKA). Penyakit jantung koroner stabil tanpa gejala biasanya diketahui dari skrining, sedangkan angina pektoris stabil didapatkan gejala nyeri dada bila melakukan aktivitas yang melebihi aktivitas sehari-hari [2].

Penyakit jantung atau yang disebut juga koroner adalah penyakit dengan kondisi dimana adanya timbunan lemak di pembuluh darah arteri koroner pada jantung dan adanya hambatan aliran darah menuju jantung [3]. Faktor utama penyebab penyakit jantung yang tidak dapat diubah, diantaranya, keturunan, usia, dan jenis kelamin, sedangkan faktor utama yang bisa dirubah adalah kebiasaan merokok, pola makan, tingginya gula darah, kolesterol, tekanan darah tinggi, dan obesitas [4]. Faktor lain gejala yang dapat didiagnosa sebagai penyakit jantung adalah sakit dada, nilai tes EKG (*resting electrodiagraphic*

"restag")), denyut jantung dan kadar gula dalam darah [5].

Salah satu jenis penyakit jantung yang berbahaya yaitu penyakit jantung koroner stabil tanpa gejala [2]. Hal ini membutuhkan suatu metode untuk membantu langkah awal penangananya, yaitu Prediksi. Hasil prediksi penyakit jantung dapat diterapkan dan digunakan mendukung keputusan tindakan apa yang harus dilakukan oleh seorang dokter atau tenaga ahli kesehatan sebagai alat bantu dalam penentuan penyakit jantung dan langkah awal penanganannya [6]. Prediksi ini menggunakan data yang cukup dan akurat yang akan menghasilkan suatu informasi yang memiliki makna, sehingga dapat menjadi sebuah pengetahuan yang lebih dikenal dengan data mining.

Data mining adalah proses mencari pola atau informasi bermakna dalam data terpilah dengan mengunakan suatu metode atau teknik tertentu. Pemilihan metode atau algoritma yang tepat sangat bergantung pada tujuan dan proses *Knowledge Discovery in Database* (KDD) secara keseluruhan [7]. Data mining dapat dibedakan berdasarkan pola, diantaranya estimasi, prediksi, klasifikasi, klustering, dan asosiasi [8].

Beberapa penerapan metode prediksi penyakit jantung telah diusulkan, diantaranya prediksi terjangkitnya penyakit jantung dengan metode *learning vector quantization* dengan hasil akurasi sebesar 66.79% [9], *analisis* dan prediksi penyakit jantung koroner dengan menggunakan metode pengenalan pola dari data catatan rekam medis penderita penyakit jantung koroner yang ada di kota Ambon periode 2014-2015 menggunakan jaringan syaraf tiruan (*backpropagation*), pengujian ini didapatkan hasil konfigurasi terbaik dari 5 tipe yang diuji dengan 13 *input* dari 6 *hidden layer* dengan 1 *output* (13-6-1) [10]. Penelitian selanjutnya menggunakan algoritma *neural network backpropagation* untuk penyakit jantung dengan menggunakan 14 dari 76 atribut untuk didapatkan hasil nilai *accuracy, precision, recall* dan AUC dari algoritma. Dari hasil uji coba dengan 572 data set telah didapatkan hasil bahwa nilai *accuracy* sebesar 91.45%, *precision* sebesar 92.79%, dan nilai AUC sebesar 0.937 yang dapat disimpulkan bahwa pemecahan untuk prediksi penyakit jantung lebih akurat [11]. Prediksi penyakit jantung dengan algoritma *classification* dan *regression tree* (CART) dengan menggunakan 8 atribut dihasilkan hasil akurasi sebesar 77% [12]. Prediksi penyakit jantung dilakukan dengan 13 atribut yang digunakan dari data set untuk membandingkan hasil eksperimen menggunakan metode ke-1 *support vector machine (SVM)* dengan metode ke-2 *support machine (SVM)* berbasis *particle swarm optimization*. Metode ke-1 didapatkan hasil akurasi sebesar 81.85% dan nilai AUC sebesar 0.899, sedangkan pada eksperimen ke-2 mengalami peningkatan hasil prediksi sebesar 88.61% dan nilai AUC nya sebesar 0.919, sehingga disimpulkan bahwa metode SVM berbasis *particle swarm optimization* lebih akurat [13]. Penelitian selanjutnya, dengan menggunakan 14 atribut dari data set yang sama pada penelitian [11], dihasilkan bahwa lima (5) faktor paling penting atau dikenal *feature importance* dari 14 atribut yang mempengaruhi penyakit jantung dari semua usia dan jenis kelamin, yaitu nyeri dada (*chest pain)*, cacat tetap atau jantung tanpa cacat *(fixed defect or heart without defect)*, jumlah pembuluh darah (*number of vesseles), exercise induced angine*, dan *slopes of the peak exercise ST segment* [14].

Hasil *feature importance* yang telah diketahui dari hasil eksperimen atau percobaan merupakan hasil yang ingin dilihat *feature* yang paling mempengaruhi sebagai hasil yang paling penting dalam menentukan prediksi penyakit jantung dan penyakit jantung sering menjangkit pada anak-anak, orang dewasa, dan tetap menjadi masalah utama di negara-negara berkembang [15].

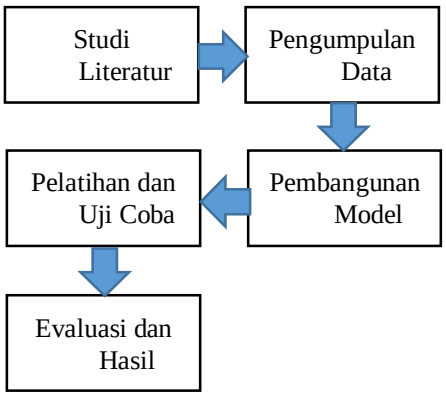
Dari hasil literatur yang telah dilakukan, secara umum algoritma yang digunakan untuk memprediksi adalah *neural network*. Penelitian ini akan melakukan prediksi awal penyakit jantung dengan menggunakan algoritma *neural network backpropagation* dan menambahkan metode LBFGS serta SGD untuk mengoptimalkan performa algoritma *neural network*, sehingga menghasilkan tingkat akurasi prediksi yang tinggi. Optimasi pada *neural*

network backpropagation adalah optimasi numerik interatif untuk mencari titik yang meminimumkan suatu fungsi yang dapat diturunkan [16]. *Stochastic Gradient Descent* (SGD) merupakan salah satu algoritma atau metode optimasi *default* dari *backpropagation* untuk mengubah nilai bobot dalam iterasi (*epoch*) untuk meminimumkan nilai fungsi. Algoritma optimasi lain yang terkenal digunakan adalah algoritma quasi newton LBFGS (*limited-memory broyden-fletcher-goldfarb-shanno)*. Metode LBFGS merupakan salah satu algoritma quasi-Newton yang terkenal untuk optimisasi yang tidak dibatasi [17].

Berdasarkan uraian dari literatur review yang telah dilakukan, penelitian ini bertujuan untuk mengetahui nilai akurasi, *precision*, *recall* dan *f1-score* untuk memprediksi penyakit jantung dengan 14 atribut *feature importance* dengan menggunakan algoritma *neural network backpropagation* dan optimasi algoritma LBFGS dengan SGD.

METODE PENELITIAN

Metode penelitian merupakan langkah dan prosedur yang dilakukan guna memecahkan permasalahan dan menguji hipotesis penelitian [18]. Adapun metode penelitian yang digunakan dalam penelitian ini seperti pada gambar 1 berikut:

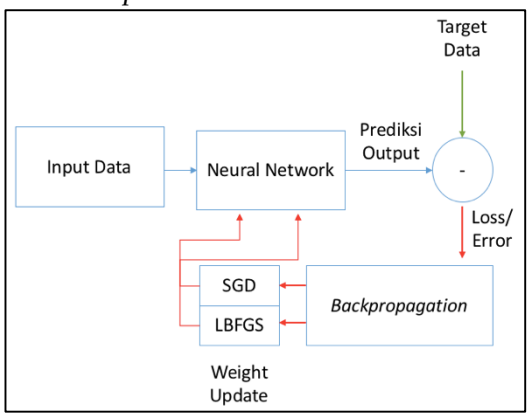


Gambar 1. *Metode Penelitian*

Studi literatur yang digunakan dalam penelitian ini digunakan untuk menghimpun data-data dan sumber referensi yang dijadikan acuan dengan keterhubungan topik yang diangkat. Studi literatur yang didapat dari berbagai sumber, seperti jurnal ilmiah, artikel, buku maupun portal berita ilmiah mengenai *backpropagation, stochastic gradient descent,* dan LBFGS. Setelah itu dilanjutkan dengan pembangunan model yang digunakan sebagai prototipe untuk disimpan sebagai memori dan menyelesaikan masalah untuk digunakan fase pelatihan, dimana data dilatih dan dianalisis menggunakan algoritma, sehingga model pembelajaran dapat direpresentasikan dalam bentuk aturan tertentu. Selanjutnya, dilakukan ujicoba untuk mendapatkan hasil dan dievaluasi.

Algoritma *Backpropagation*

Backpropagation merupakan model algoritma yang melibatkan tiga lapisan, yaitu lapisan input, data diperkenalkan ke jaringan menuju lapisan tersembunyi (*hidden layer*), tempat data diporses dan hasil (*output*) layer, dimana hasil input yang diberikan atau dikirimkan diproduksi untuk mendapatkan hasil [19]. Algoritma ini dimaksudkan untuk menyesuaikan (*mengupdate*) kembali setiap bobot (*weight*) dan bias dengan *learning rate* tertentu berdasarkan error yang didapat pada saat *forward pass*. *Backrpropagation* merupakan tahap menghitung gradien dari *loss function* terhadap semua parameter yang ada dengan cara mencari nilai fungsi turunan dengan metode optimasi SGD dengan LBFGS. Posisi algoritma *backpropagation* dapat dilihat pada gambar 2. Tanda panah biru menunjukkan *Forward pass* dan panah merah menunjukkan *backward pass*.



Gambar 2. *Alur Pelatihan Algoritma Neural Network*

Dalam *supervised learning*, pelatihan data terdiri dari input dan output (target). Pada tahap input data, data akan di *propagate* (*forward pass*) menuju output layer dan hasil prediksi output akan dibandingkan dengan target dengan menggunakan sebuah fungsi yang disebut *Loss Function: Binary Cross-Entropy/Log Loss*. Fungsi ini digunakan untuk mengukur seberapa baik atau bagus performa dari neural network dalam melakukan prediksi terhadap target [20]. Perbandingan ini dapat dengan menggunakan persamaan 1 berikut.

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{n} y_i \bullet log(p(y_i)) + (1 - y_i) \bullet log(1 - p(y_i)) \quad (1)$$

Dimana y adalah label (1 untuk probabilistik bernilai *true* atau *yes*, dan 0 benrilai *false* atau *no*) dan p(y) prediksi probabilitas nilai bernilai 1 untuk semua nilai N. Formula 1 menunjukkan bahwa setiap yang bernilai *true*, maka (y = 1), ini menambahkan *log (p(y))* kerugian (*loss*), yaitu probabilitas *log* bernilai *true*. Sebaliknya, ini menambahkan *log (1-p(y))*, yaitu probablitas *log* untuk bernilai *false* (y = 0).

Langkah pelatihan dengan *backpropagation* terdapat 3 langkah utama. Fase pertama adalah fase *forward pass* yang dimana pila masukan dihitung maju mulai dari layar masukan hingga layar keluaran menggunakan fungsi aktivasi. Fase kedua adalah *backward pass*, dimana selisih antara keluaran jaringan dengan target yang diinginkan yang disebut kesalahan (*loss*). Kesalahan ini dipropagasikan mundur, dimulai dari garis yang berhubungan langsung dengan *neuron* di layar keluaran. Fase ketiga yaitu perubahan nilai bobot untuk menurunkan kesalahan yang terjadi. Berikut ini langkah-langkah *backpropagation* [21] [22]:

Fase 0: Awal

Langkah 1: Inisialisasi nilai bobot dengan bilangan acak.

Langkah 2: Jika kondisi penghentian belum terpenuhi, dilakukan iterasi sampai *epoch* terpenuhi dari langkah 3 – 9.

Langkah 3: Setiap pasang data pelatihan, lakukan langkah 4-9.

Fase 1: *Forward Pass*

Langkah 4: Setiap data unit masukkan menerima sinyal dan meneruskan ke *neuron* pada *hidden layer*

Langkah 5: Menghitung semua keluaran di *neuron* pada *hidden layer* dengan Z_j (j = 1, 2, 3, … p): dengan persamaan (2).

$$Z_{pathj} = V_{j0} + \sum_{i=1}^{n} X_i V_{ji} \quad (2)$$

Langkah 6: Menghitung semua jalur (*path*) di unit keluaran (Y_k), seperti pada persamaan (3).

$$Y_{pathk} = W_{k0} + \sum_{j=1}^{p} Z_i W_{kj} \quad (3)$$

Fase 2: *Backward Pass*

Langkah 7: Hitung faktor δ unit keluaran berdasarkan kesalahan (*lost*/*error*) di setiap unit keluaran y_k (k = 1, 2, 3, …, m).

$$\delta_k = (t_k - y_k)\, f'' (y_{pathk}) = (t_k - y_k)\, y_k(1 - y_k) \quad (4)$$

Langkah 8: Menghitung penjumlahan keslaahn (*lost/error*) dari unit tersembunyi di *hidden layer* (= δ).

$$\delta_{pathj} = \sum_{k=1}^{m} \delta_k W_{kj} \quad (5)$$

Fase 3: Perubahan Bobot (*Weight Update*)

Pada fase ini dilakukan perhitungan nilai semua perubahan bobot dengan optimasi yaitu SGD dan LBFGS.

A. Perubahan Bobot dengan Optimasi SGD

Stochastic Gradient Descent (SGD) *update* merupakan algoritma yang digunakan untuk mengupdate parameter *weight* dan bias [17].

$$W_{kj}\ (new) = W_{kj}\ (old) + \Delta W_{kj} \quad (6)$$

Perubahan bobot (*weigh*) unit tersembunyi:

$$V_{ji}\ (new) = V_{ji}\ (old) + \Delta V_{ji} \quad (7)$$

B. Perubahan Bobot dengan Optimasi LBFGS

Algoritma ini merupakan salah satu metode quasi-Newton yang efisien untuk optimisasi *unconstrained*. Algoritma ini dikenalkan oleh Broyden, Fletcher, Goldfarb, dan Shanno [17], dan dikembangkan dengan pembaaruan *limited storage* yang digunakan untuk quasi-Newton metriks untuk meminimkan kondisi awal dalam menyelsaikan masalah dimana memori penyimpanan merupakan hal yang penting [23]. Berikut ini algoritma algoritma dasar LBFGS [17] [23]:

Min $f(\chi)$, $\chi \in R^n$ (8)

Langkah 0.
Berikan nilai $\chi_1 \in R^n$; $B_1 \in R^{n \times n}$; bilangan mutlak positif; (9)

Menghitung $g_1 = \Delta f(\chi_1)$. Jika $g_1 = 0$, *berhenti;* Jika tidak, atur $k := 1$. (10)

Langkah 1. Atur $d_k = - B_k^{-1} g_k$ (11)

Langkah 2. Lakukan pencarian baris selama d_k, dapatkan $\alpha_k > 0$,
$\chi_{1+1} = \chi_k + \alpha_k d_k$, dan $g_{k+1} = \Delta f(\chi_{1+1})$;
Jika dan $g_{k+1} = 0$, berhenti (12)

Langkah 3.

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{s_k^T y_k}, \quad (13)$$

dimana

$s_k = \alpha_k d_k$ (14)

$y_k = g_{k+1} - g_k$ (15)

Langkah 4.
$K := k + 1$; pergi ke Langkah 1. (16)

Penelitian ini merupakan penelitian eksperimen dengan menggunakan data set sekunder yang telah disediakan oleh *Center for Machine Learning and Intelligent Systems* [24]. Penelitian ini menggunakan 303 data set sekunder yang terdiri dari 165 record atribut label sakit dan 138 record atribut label sehat, dengan 14 atribut (*features*), diantaranya:

Tabel 1. *Informasi Atribut*

No	Atribut	Keterangan
1	Age (Usia)	-
2	Sex	0 = Female 1 = Male
3	ChestPain	Typical Asymptotic Nonanginal Nontypical
4	RestBP	Rest Blood Pressure
5	Chol	Serum Kolesterol dalam mg/dl
6	Fbs	Fasting blood sugar > 120 mg/dl 0 = False 1 = True
7	Restecg	Resting Electrocardiographic result
8	MaxHR	Maximum heart rate achieved
9	ExAng	Exercise induced angine 0 = no 1 = yes
10	Oldpeak	ST depression induced by exercise relative to rest
11	Slope	Slope of the peak exercise ST segment
12	Ca	Number of major vessels colored by flourosopy (0-3)
13	Thal	3 = normal 6 = fixed defect 7 = reversible defect
14	Target	AHD – Diagnosis heart disease 0 = no 1 = yes

HASIL DAN PEMBAHASAN

Pelatihan dan pengujian pada penelitian ini menggunakan Tensorflow bahasa pemrograman Python. Langkah pertama yang dilakukan adalah dengan mengimport atau mengecek path data set terlebih dahulu dan ditampilkan dengan kode program pada gambar 3 berikut.

```
df = pd.read_csv('../input/heart-
    disease/heart.csv')
df.head(20)
```

Gambar 3. *Kode Program Cek Path Dataset*

Kode program pada gambar 3 ditunjukkan untuk mengambil path data set yang ada, kemudian ditampilkan sebanyak 20 record atau baris. Hal ini dilakukan untuk mengetahui apakah data sudah berhasil diambil dan bisa digunakan seperti pada gambar 4.

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
5	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1
6	56	0	1	140	294	0	0	153	0	1.3	1	0	2	1
7	44	1	1	120	263	0	1	173	0	0.0	2	0	3	1
8	52	1	2	172	199	1	1	162	0	0.5	2	0	3	1
9	57	1	2	150	168	0	1	174	0	1.6	2	0	2	1
10	54	1	0	140	239	0	1	160	0	1.2	2	0	2	1
11	48	0	2	130	275	0	1	139	0	0.2	2	0	2	1
12	49	1	1	130	266	0	1	171	0	0.6	2	0	2	1
13	64	1	3	110	211	0	0	144	1	1.8	1	0	2	1
14	58	0	3	150	283	1	0	162	0	1.0	2	0	2	1
15	50	0	2	120	219	0	1	158	0	1.6	1	0	2	1
16	58	0	2	120	340	0	1	172	0	0.0	2	0	2	1
17	66	0	3	150	226	0	1	114	0	2.6	0	0	2	1
18	43	1	0	150	247	0	1	171	0	1.5	2	0	2	1
19	69	0	3	140	239	0	1	151	0	1.8	2	2	2	1

Gambar 4. *Data Set Ditampilkan 20 Record*

Langkah berikutnya yaitu mengecek apakah terdapat nilai (*value*) yang hilang (kosong) pada data set yang digunakan, jika terdapat nilai yang hilang maka harus dilakukan normalisasi, jika tidak maka, data set dapat digunakan. Pada penelitian ini, nilai pada data set sudah terisi semua (*false*) atau tidak ada yang hilang. Dalam melakukan pengecekan nilai seperti pada gambar 5.

```
assert df.dtypes.any() != object
df.isnul().value.any()
```

```
False
```

Gambar 5. *Hasil Cek Nilai yang Hilang*

Pada 14 atribut yang digunakan terdapat lima (5) atribut yang bertipe kategorial, maka dibutuhkan transformasi nilai. Lima atribut ini diantaranya *Chestpain (cp), ca, thal, slope,* dan *restecg*. Pada python menggunakan *one-hot encoding*. Proses ini dimana fitur kategorial dikonversi ke dalam bentuk data numerikal yang dapat diproses atau didukung untuk algoritma dalam melakukan pekerjaan yang lebih baik dalam prediksi, seperti array numerik. Input ke transformator ini berupa array, seperti integer atau string yang menunjukkan nilai yang diambil oleh fitur ketagorial. *One-hot encoding* [25]. Kode program yang digunakan seperti pada gambar 6.

```
def encode(data, col):
    return pd.concat([data, pd.get_dummies(col, prefix=col.name)], axis=1)
df = encode(df, df.cp)
df = encode(df, df.thal)
df = encode(df, df.slope)
df = encode(df, df.restecg)
df = encode(df, df.ca)

df_new = df[['cp', 'thal', 'slope', 'restecg', 'ca']].copy()
df_new = encode(df_new, df.cp)
df_new = encode(df_new, df.thal)
df_new = encode(df_new, df.slope)
df_new = encode(df_new, df.restecg)
df_new = encode(df_new, df.ca)

df.drop(['cp', 'thal', 'slope', 'restecg', 'ca'], axis=1, inplace=True)
df_new.drop(['cp', 'thal', 'slope', 'restecg', 'ca'], axis=1, inplace=True)
df.head(2)
```

Gambar 6. *Kode Program One-Hot Encoding*

Dari hasil kode program pada gambar 6, telah berhasil atribut yang kategorial diubah menjadi sebuah array, sehingga kolom pun bertambah menjadi 28 kolom dari 14 kolom awal karena menggunakan 14 atribut yang digunakan. Contoh hasil seperti pada gambar 7.

	age	sex	trestbps	chol	fbs	thalach	exang	oldpeak	target	cp_0	...	slope_1	slope_2	restecg_0	res
0	63	1	145	233	1	150	0	2.3	1	0	...	0	0	1	0
1	37	1	130	250	0	187	0	3.5	1	0	...	0	0	0	1

2 rows × 28 columns

Gambar 7. *Bertambahnya Jumlah Kolom Hasil One-Hot Encoding*

Selanjutnya, disiapkan data untuk pelatihan dengan membagi atau memisahkan (*split*) data dari data set yang digunakan. Kode program untuk pelatihan dan uji coba ini digunakan untuk melatih metode dan membagi data set untuk pelatihan dan uji coba. Berdasarkan kode program seperti pada gambar 8, sistem menggunakan 242 dengan 27 *neuron* pada *hidden layer* untuk pelatiha, seperti pada gambar 9.

```
From sklearn.model_selection
   import train_test_split

xTrain_14, xTest_14, yTrain_14,
   yTest_14 =
   train_test_split(df, target,
   test_size =0.2, random_state
   =0)
```

Gambar 8. *Kode Program Training dan Test*

Gambar 9. *Hasil Membagi Data Training dan Test*

Langkah berikutnya yaitu membuat kode program untuk pelatihan, percobaan, dan pengujian prediksi penyakit jantung dengan algoritma *neural network backpropagation* dengan membandingkan optimasi 2 (dua) metode yaitu LBFGS dan SGD. Penelitian ini menggunakan *neuron* pada *hidden layer* sebanyak 15 (default) dan 20 batch size yang berarti setiap inputan maksimal ada 20 data dalam 1 kali iterasi (*epoch*). Kode program seperti pada gambar 10.

```
from sklearn.neural_network import MLPClassifier

logits_lbfgs = MLPClassifier(random_state=123, solver='lbfgs',
 hidden_layer_sizes=(15,), batch_size=20)
logits_sgd = MLPClassifier(random_state=123, solver='sgd', hid
den_layer_sizes=(15,), batch_size=20)
```

Gambar 10. *Kode Program Algoritma backpropagation dan Optimasi Metode LBFGS dan SGD*

Selanjutnya, dilakukan pelatihan (*training*) terhadap data set dengan menggunakan algoritma dan masing-masing optimasi. Berdasarkan hasil pelatihan, *CPU Times* dan *Wall Time* dengan optimasi LBFGS lebih baik dibandingkan SGD. *CPU Times* digunakan untuk mengukur waktu selama prosesor bekerja secara aktif pada tugas tertentu, yaitu pelatihan data, sedangkan *Wall Time* digunakan untuk mengukur total waktu untuk menyelesaikan proses sampai selesai. Hasil waktu pelatihan data seperti pada tabel 2 berikut ini.

Tabel 2. *Hasil Waktu Pelatihan Data*

No	Metode Optimasi	Hasil
1	LBFGS	CPU Times: 46.5 ms Wall Time: 54.1 ms
2	SGD	CPU Times: 908 ms Wall Time: 909 ms

Langkah berikutnya mengecek *loss/error* dari metode optimasi yang digunakan dengan menggunakan kode program pada gambar 11 dan dihasilkan LBFGS sebesar 0.0011, sedangkan SGD memiliki tingkat *loss/error* lebih tinggi, yaitu sebesar 0.5468. Hasil perhitungan *Loss/Error* ini digunakan untuk mengukur kinerja model yang keluaran (*output*)-nya merupakan nilai probabilitas antara 0 dan 1. *Loss Binary Cross-Entropy* meningkat ketika probabilitas yang diprediksi menyimpang dari label aktual atau dengan kata lain untuk mengukur besar perbedaan antara hasil prediksi model dan target keluaran (*output*)-nya [26]. Hasil seperti pada tabel 3.

```
print("final LBFGS score: ",
   hist_lbfgs.loss_)

print("final SGD score: ",
   hist_SGD.loss_)
```

Gambar 11. *Kode Program Menghitung Loss/Error*

Tabel 3. *Hasil Loss/Error*

No	Metode Optimasi	Hasil
1	LBFGS	0.001152
2	SGD	0.546822

Selanjutnya, dilakukan prediksi akurasi untuk penyakit jantung dengan algoritma *neural network backpropagation* dengan metode optimasi LBFGS dan SGD.

```
yPred_lbfgs                              =
    logits_lbfgs.predict(xTest_14)

yPred_sgd                                =
    logits_sgd.predict(xTest_14)

from      sklearn.metrics      import
    classification_report,
    accuracy_score

print(classification_report(yTest_14,
    yPred_lbfgs))
print(accuracy_score(yTest_14,
    yPred_lbfgs))

print(classification_report(yTest_14,
    yPred_sgd))
```

Gambar 12. *Kode Program Prediksi Penyakit Jantung*

Dalam hasil prediksi ini terdapat empat parameter yang dihasilkan yaitu, *accuracy*, *precision*, *recall*, dan *F1-Score*. *Accuracy* atau akurasi digunakan untuk mengukur kinerja yang paling intuitif, dalam hal ini yaitu rasio pengamatan yang diprediksi dengan benar terhadap total pengamatan dari data set untuk model penelitian ini. Secara umum, jika akurasi tinggi, maka model juga baik. Disamping itu, untuk mengevaluasi kinerja model ini, harus memperhatikan parameter lainnya, yaitu *Precision*, *Recall*, dan *F1-Score*. Presisi (*Precision*) adalah rasio pengamatan positif yang diprediksi dengan benar terhadap total pengamatan positif yang diprediksi. Presisi merupakan seberapa tepat/akurat model dari yang diprediksi positif dan berapa banyak hasil dari yang sebenarnya positif, misalnya bahwa diberikan metrik bahwa dari semua pasien yang diberikan label sebagai positif penyakit jantung, berapa banyak yang sebenarnya yang positif jantung. Presisi tinggi berkaitan dengan tingkat *low false positive*. Sedangkan *recall* merupakan rasio pengamatan positif yang diprediksi dengan benar dengan semua pengamatan di kelas (*class*) aktual *yes* atau *true*, dan F1-Score merupakan rata-rata dari *precision* dan *recall*. F1-Score biasanya lebih bermanfaat daripada akurasi, terutama jika memiliki distribusi kelas yang tidak merata. Akurasi bekerja paling baik jika *false positive* dan *false negative* memiliki biaya *(cost*) yang sama [27][28]. Berdasarkan kode program pada gambar 12, dihasilkan seperti pada tabel 4 berikut ini.

Tabel 4. *Hasil Prediksi Penyakit Jantung dengan Metode Optimasi*

Hasil	LBFGS	SGD
Accuracy	85.3%	83.6%
Precision	86%	84%
Recall	84%	86%
F1-Score	85%	85%

PENUTUP

Berdasarkan hasil dan pembahasan yang didapat, prediksi penyakit jantung dengan menggunakan algoritma *neural network backpropagation* dengan metode optimasi LBFGS lebih baik dibandingkan dengan SGD. Hal ini ditunjukkan dengan hasil eksperimen bahwa tingkat *accuracy* sebesar 85.3% dibandingkan dengan metode SGD sebesar 83,6%. Hal lain ditunjukkan dengan hasil *lost/error* bahwa metode optimasi LBFGS memiliki hasil error mendekati 0, yaitu sebesar 0.001152 dan waktu proses *CPU Times* hanya membutuhkan waktu 46.5 ms dibandingkan SGD selama 908 ms. Disamping itu, hasil *wall time* dengan optimasi LBFGS sebesar 54.1 ms, sedangkan optimasi SGD sebesar 909 ms. Hal ini menandakan bahwa prediksi penyakit jantung menggunakan algoritma *neural network* dengan optimasi LBFGS lebih baik dibandingkan SGD.

DAFTAR PUSTAKA

[1] Direktorat Jenderal PP & PL, Kemenkes RI, "Pedoman Pengendalian Faktir Risiko Penyakit Jantung Dan Pembuluh Darah", Edisi 1, pp. 4, 2011. Tersedia di: http://www.p2ptm.kemkes.go.id/dokumen-ptm/pedoman-pengendalian-faktor-risiko-penyakit-jantung-dan-pembuluh-darah

[2] P2PTM Kemenkes RI, "Hari Jantung Sedunia (HJS) Tahun 2019: Jantung Sehat, SDM Unggul", 2019, Tersedia di: http://p2ptm.kemkes.go.id/kegiatan-

p2ptm/pusat-/hari-jantung-sedunia-hjs-tahun-2019-jantung-sehat-sdm-unggul

[3] S. C. Smletzer, B. G. Bare., *Keperawatan Medikal Bedah Brunner & Suddarth Edisi 8-vol 2* Jakarta: EGC. 2013.

[4] P. W. Wahyuni, C. H. Rosjidi, S. Nurhidayat, "Identifikasi usia Sebagai Faktor Risiko Penyakit Jantung Koroner Pada Perempuan Di Poli Jantung RSUD Dr. Harjono Ponorogo", Health Science Journal Vol 3, No.1 ISSN 2598-1196(Online), April, 2019.

[5] A. M. Mahmood, M. R. Kuppa, "Early Detection Tree Algorithm", *Second Vaagdevi Internasional Conference on Information Technology for Real World Problems,* pp. 24-28, 2011.

[6] R. A. Pramunendar, I. N. Dewi, H. Asari, "Penentuan Prediksi Awal Penyakit jantung Menggunakan Algoritma Back Propagation Neural Network dengan Metode Adaboost", Seminar Nasional Teknologi Informasi & Komunikasi Terapan (SEMANTIK), ISBN: 979-26-0266-6, pp. 298-304, 2013.

[7] Y. Mardi, "Data Mining : Klasifikasi Menggunakan Algoritma C4.5", Jurnal Edik Informatika Penelitian Bidang Komputer Sains dan pendidikan Informatika V2.i2, E-ISSN : 2541-3716, pp. 213-219, 2017.

[8] D. T. Larose, "*Discovering Knowledge in Data",* New Jersey: John Willey & Sons, Inc, 2005.

[9] N. Hidayati, B. Warsito, "Prediksi Terjangkitnya Penyakit Jantung Dengan Metode Learning Vector Quantization", Media Stastistika, Vol. 3, No. 1, pp 21-30, Juni, 2010.

[10] D. L. Rahakbauw, F. K. Lembang, Y.M. J. Taihuttu, "Analisis Dan Prediksi Penyakit Jantung Koroner Di Kota Ambon Menggunakan Jaringan Syaraf Tiruan", Barekeng: Jurnal Ilmu Matematika dan Terapan Vol. 10 No. 2, 2016.

[11] B. Rifai, "Algoritma Neural Network Untuk Prediksi Penyakit Jantung", Jurnal Techno Nusa Mandiri Vol. IX No. 01, Maret, 2013.

[12] S. Nurajizah, "Penerapan Metode *Support Vector Machine* Berbasis *Particle Swarm Optimization* Untuk Prediksi Penyakit Jantung", Jurnal Techno Nusa Mandiri, Vol. X No.1, pp. 216-226, September, 2013.

[13] N. A. Puteri, W. Maharani, M. D. Suliiyo, "Prediksi Penyakit Jantung Dengan Algoritma Classification dan Regression Tree", Tugas Akhir, Fakultas Teknik Informatika, Telkom University, 2013.

[14] Volody, "Heart Disease Daat Exploration", Januari 2020, Tersedia: https://www.kaggle.com/volodymyrgavrysh/hear-disease-data-exploration

[15] I. Y. Hananta dan H. F. Muhammad, "Dietisian Deteksi Dini & Pencegahan 7 Penyakit Penyebab Mati Muda", Yogyakarta: Media Pressindo, 2011.

[16] K. Kurniawan, "Bagaimana menjelaskan Stochastic Gradient Descent dan kaitannya dengan regresi linear", S2 Kecerdasan Buatan, University of Edinburg, 2017. Tersedia di: https://id.quora.com/Bagaimana-anda-menjelaskan-Stochastic-Gradient-Descent-dan-kaitannya-dengan-regresi-linear

[17] Yu-Hong Dai, "Convergence properties of The BFGS Algorithm", SIAM J. OPTIM. Vol. 13, No. 3, pp. 693-701, 2002.

[18] A. Wanto, "Analisis Jaringan Syaraf Tiruan Backpropagation Menggunakan Conjugate Gradient Fletcher Reeves Dalam Proses Memprediksi, "Tesis, Universitas Sumatera Utara, 2017.

[19] D. Huang dan Z. Wu, "Forecasting outpatient visits using empirical mode decomposition coupled with backpropagation artificial neural networks optimized by particle swarm

optimization, "PLoS ONE, Vol. 12, No. 2, pp. 1-18, 2017.

[20] D. Godoy, "Understanding binary cross-entropy log loss: a visual explanation", towards data science, 22 Nov, 2018. Tersedia di: https://towardsdatascience.com/understanding-binary-cross-entropy-log-loss-a-visual-explanation-a3ac6025181a

[21] S. Sena, "Pengenalan Deep Learning Part 3: Backpropagation Algorithm", Medium, 3 Nov, 2017. Tersedia di: https://medium.com/@samuelsena/pengenalan-deep-learning-part-3-backpropagation-algorithm-720be9a5fbb8

[22] A. Wanto, "Optimasi Prediksi Dengan Algoritma Backpropagation Dan Beale-Powell Restarts", Jurnal Nasional Teknologi dan Sistem Informasi, Vol. 03, No. 03, pp. 370-380, 2017.

[23] J. Nocedal, "Updating Quasi-Newton Matrices With Limited Storage", Mathematics of Computation, Volume 35 Number 151, pp. 773-782, July, 1980.

[24] A. Janosi, W. Steinbrunn, M. Pfisterer, R. Detrano, "Heart Disease Data Set", Center for Machine Learning and Intelligent Systems, Tersedia di: https://archive.ics.uci.edu/ml/datasets/Heart+Disease

[25] Scikit-learn, "Sklearn.preprocessing.OneHotEncoding", 2019, Tersedia di: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html

[26] ML Glossary, "Loss Function", 2017, Tersedia di: https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html

[27] Renuka Joshi, "Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures", 2016, Exsilio Solutions, Tersedia di: https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/

[28] Koo Ping Shung, "Accuracy, Precision, Recall or F1?", Towards Data Science, 2018, Tersedia di: https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9