IMPLEMENTASI *NEURAL NETWORK* UNTUK MEMPREDIKSI JUMLAH PENDERITA *TUBERCULOSIS*

Ahmad Chamsudin

Program Studi Pendidikan Teknik Informatika
Fakultas Keguruan dan Ilmu Pendidikan, Universitas Muhammadiyah Surakarta
Jl. A. Yani Tromol Pos I Pabelan, Kartasura, Surakarta, Indonesia
Ahmad.chamsudin@ums.ac.id

Abstract — *Tuberkulosis (TBC) merupakan salah satu jenis penyakit menular yang memiliki jumlah penderita yang sangat banyak, bahkan Indonesia menduduki urutan ke empat jumlah terbanyak penderita Tuberkulosis di dunia. Tuberkulosis telah banyak menyebabkan kematian pada penderitanya baik dari kalangan anak-anak, dewasa sampai lanjut usia. Dengan fenomena tersebut maka dibutuhkan studi untuk memprediksi jumlah penderita tuberkulosis pada tahun-tahun yang akan datang yang nantinya dapat digunakan sebagai pendukung keputusan medis. Dalam penelitian ini Neural network digunakan untuk memprediksi jumlah penyakit tuberkulosis dengan menggunakan data bulanan dalam jangka waktu duapuluh tahun terakhir. Metode yang digunakan untuk membangun neural network yaitu dengan menggunakan tiga macam algoritme yaitu Back Propagation, Quasi-Newton dan Lavemberg-Marquardt untuk dapat meminimalkan tingkat error, dengan harapan hasil yang dicapai bisa lebih tepat dalam memprediksi jumlah penyakit tuberkulosis. Dalam penelitian ini diperoleh hasil forecasting dengan mengukur tingkat akurasi forecasting dengan membandingkan nilai MAPE dan MSE, dari ketiga algoritme didapatkan algoritme Lavemberg-Marquardt memiliki nilai MAPE dan MSE terbaik yaitu 9,15 % dan 43419 yang menunjukkan algoritme yang paling optimal dibanding yang lain.*

Keywords— Neural Network, Forecasting, Tuberculosi, Back Propagation, Quasi-Newton,Lavemberg-Marquardt.

I. PENDAHULUAN

Penyakit *Tuberkulosis* (TBC) merupakan salah satu penyakit menular yang sangat mematikan, dimana di Indonesia berdasarkan catatan USAID (*The United States Agency for International Development*), merupakan salah satu penyumbang penyakit TBC nomor empat di dunia setelah India, China dan Afrika Selatan. Diperkirakan ada 430 ribu kasus TBC baru dan 169 orang di antaranya meninggal setiap hari. Angka kematian perempuan akibat TBC jauh lebih tinggi dibandingkan dengan kematian lantaran persalinan. Penyakit menular ini juga menjadi penyebab utama kematian perempuan. Kementerian Kesehatan RI mencatat, angka kematian ibu akibat persalinan 10.488 per tahun atau 228 per 100.000 kelahiran hidup. Sementara itu, kematian akibat TBC mencapai 31.873 per tahun.

Forecasting atau peramalan merupakan kebutuhan yang penting dalam kehidupan sehari-hari,baik untuk meramalkan cuaca, penyakit, memprediksi gempa bumi, memprediksi berapa banyaknya jumlah mahasiswa, dan lain-lain. Seiring dengan banyaknya bidang yang memerlukan peramalan yang lebih akurat, maka metode peramalan banyak dikembangkan oleh para peneliti.

Metode peramalan yang banyak dikembangkan oleh peneliti tersebut digunakan untuk mengetahui bagaimana pergerakan dari suatu data. Beberapa metode peramalan yang banyak dikembangkan oleh peneliti, metode peramalan kuantitatif dapat dibagi menjadi dua jenis model peramalan yang utama, yaitu metode kausal (*regresi*) dan metode *time series*[1]. Dari kedua metode tersebut yang sering digunakan adalah metode *time series*. Pemodelan *time series* yang berguna untuk kebutuhan peramalan berbagai macam kasus, memicu munculnya berbagai

penelitian ilmiah yang berkaitan dengan analisis *time series*.

Peramalan penyakit TB telah banyak terus dikembangkan dengan beberapa metode dan model untuk mendapakkan model yang optimal dalam melakukan peramalan diantara dengan membandingkan motode *SARIMA* dengan model *Hybrid* dengan model *time series* [2]. Selain itu Metode *ARIMA* juga digunakan untuk melakukan peramalan penyakit TBC [3, 4], metode yang lain yang digunakan adalah metode *Box-Jenkins* [5].

Penggunaan metode ANN untuk peramalan penyakit TB dengan menguji menggunakan tiga algoritma bersamaan untuk menentukan algoritma yang terbaik belum ada yang melakukan untuk peramalan penyakit TB atau untuk penyakit yang lainnya.

II. METODE PENELITIAN

Tujuan utama dari penelitian ini adalah untuk melakukan peramalan terhadap jumlah penderita TBC pada tahun-tahun yang akan datang berdasarkan data-data masa lampau dengan menggunakan metode *neural network*.dengan alur sebagai berikut:

1. Tahap pertama adalah pengambilan data *time series* yang kemudian dilakukan proses pengolahan data. Analisis perhitungan dilakukan menggunakan data 20 tahun terakhir dimulai dari tahun 1993-2012 dengan parameter jumlah penderita TBC setiap bulanya.
2. Tahap kedua dengan analisis data dilakukan untuk melihat bulan-bulan apa saja yang signifikan tiap tahunnya. Hal tersebut dilakukan agar dapat disesuikan dengan hasil peramalan yang nantinya didapat. Di sisi lain adalah untuk membuktikan apakah data tersebut mengandung faktor *seasonal* (musiman) atau tidak sehingga dapat ditentukan metode yang sesuai dengan masalah dan data yang ada.
3. Tahap ketiga melakukan perancangan arsitektur ANN dengan menentukan masukan, jumlah layer tersembunyi, menentukan bobot dan *output*.
4. Tahap keempat *Training* merupakan proses untuk melakukan interasi terhadap arsitektur dalam *neural network.* Dalam penelitian ini melakukan pengujian terhadap tiga macam Algoritme yaitu *Back propagation, Quasi Newton* dan *Lavemberg Marquardt*.

   a. Algoritma *Back Propagation*

   Algoritme *Back propagation* merupakan salah satu variasi dalam jaringan saraf tiruan. Perbedaannya dengan jaringan saraf tiruan yang lain ialah proses pembelajarannya dilakukan dengan penyesuaian bobot-bobot jaringan saraf tiruan dengan arah mundur dan didasarkan oleh *error* ketika proses pembelajaran[9]. *Back propagation* bekerja secara iterative dengan menggunakan sekumpulan data, dari data-data tadi dalam setiap proses yang dilakukan dicari bobot relasinya, lalu dimodifikasi agar nilai *Mean Squared Error(MSE)* antara jaringan dengan nilai sesungguhnya seminimal mungkin.

   Kinerja metode pembelajaran B*ack Propagation* dipengaruhi oleh parameter – parameternya, diantaranya adalah *learning rate* dan *momentum*. *Learning rate* merupakan salah satu pertimbangan penting dalam kinerja jaringan saraf yang ditentukan oleh bagaimana kita merubah bobot-bobot ‘w’ pada tiap langkah, jika *learning rate* terlalu kecil algoritme akan memakan waktu lama menuju konvergen dan sebaliknya jika *learning rate* terlalu besar maka algoritme menjadi divergen. Algoritme pelatihan untuk *Back propagation* dengan satu layer tersembunyi (dengan fungsi aktivasi *sigmoid biner*) sebagai berikut :

   Langkah 0: Inisialisasi bobot-bobot tetapkan dalam nilai acak kecil

   Langkah 1: Bila syarat berhenti adalah salah, kerjakan langkah 2-9.

Langkah 2: Untuk setiap pasangan pelatihan, kerjakan langkah 3-8.

Umpan maju

Langkah 3: Tiap unit masukan (I = 1 ,…. n) menerima isyarat masukan dan diteruskan ke unit-unit tersembunyi.

Langkah 4: Tiap unit tersembunyi) Menjumlahkan isyarat masukan terbobot.

∑ .....(1)

Dengan menerapkan fungsi aktivasi hitung:

).....(2)

Langkah 5: Tiap unit keluaran (y_k , k = 1 , …m) menjumlahkan isyarat masukan berbobot :

∑ .....(3)

Perambatan Balik Galat

Langkah 6: Tiap unit keluaran (y_k , k = 1 , …m) menerima pola sasaran berkaitan dengan pola pelatihan masukannya. Hitung galat informasi :

' .....(4)

Menghitung koreksi bobot dan prasikapnya :

.....(5)

.....(7)

Langkah 7: Tiap unit tersembunyi (j = 1, ….p) menjumlahkan delta masukannya (dari unit-unit lapisan atasnya)

∑ .....(8)

Hitung galat informasinya :

).....(9)

Hitung koreksi bobot prasikapnya:

.....(10)

Perbaharui bobot dan prasikap

Langkah 8: Tiap unit keluaran (y_k , k = 1 , …m) memperbaharui bobot-bobot dan prasikapnya (j = 0, 1…..,p)

.....(11)

Tiap unit tersembunyi (j = 1, ….p) memperbaharui bobot-bobot dan prasikapnya (j = 0, 1……,p)

.....(12)

Langkah 9: Uji prasyarat berhenti.

terutama dirancang untuk input *touchscreen*, juga telah digunakan dalam konsol *game* , kamera digital , dan elektronik lainnya.

b. Algoritme *Quasi Newton*

Misalkan w_t adalah vektor bobot yang mengandung $w_j(t)$, b2(t), $v_{ij}(t)$ dan $b1_j(t)$, konsep dasar metode *Newton* adalah :

$\mathbf{w_{t+1} = w_t - H_t^{-1} . g_t}$ ……...…(13)

dengan

w : vektor bobot dan bias koneksi.

$\mathbf{g_t}$: vektor gradien yang berisi $g_{j(t)}$, $g_{b2(t)}$, $g_{ij(t)}$dan $g_{b1j(t)}$.

H : matriks *Hessian*.

Matriks *Hessian* merupakan turunan kedua dari indeks kinerja terhadap bobot-bobot dan bias koneksi jaringan pada nilai ke-t, sehingga matriks *Hessian* didapatkan dengan cara yang lebih kompleks. Untuk menghindari penghitungan yang lebih kompleks tersebut matriks *Hessian* dengan memberikan inisial matriks *Hessian* pada awal pelatihan. Inisial matriks *Hessian* pada awal *epoch* harus bersifat simetrik dan definit positif.

Matriks yang biasanya digunakan sebagai inisial matriks *Hessian* adalah matriks identitas. Hal ini disebabkan matriks identitas mudah didefinisikan serta merupakan matriks simetrik dan definit positif [9]. Salah satu algoritma perubahan bobot dengan metode *newton* adalah algoritma BFGS yang diperkenalkan oleh Broyden, Fletcher, Goldfarb dan Shanno. Algoritma pelatihan dengan metode *Quasi Newton* adalah sebagai berikut [36]:

Langkah 0 :

a) Inisialisasi bobot awal dengan bilangan acak kecil
b) Inisialisasi Epoch 0, MSE ≠ 0
c) Inisialisasi H_0 yang merupakan matrik

d) definit posotif simetrik.
e) Tetapkan Maksimum *Epoch* dan Target *Error*.

Langkah 1 : Jika kondisi penghentian belum terpenuhi (*Epoch*< Maksimum *Epoch* atau MSE > Target *Error*), lakukan langkah berikutnya.

Langkah 2 : Unit output Y menerima target pola yang berhubungan dengan pola input pelatihan. Kesalahan pada unit output didefinisikan sebagai : e = (ty)

Dengan : e : Kesalahan pada unit *output*
t : Keluaran yang diinginkan (acuan/ target)
y: Keluaran aktual

Fungsi jumlah kuadrat *error* didefinisikan dengan $E=1/2\ (t-y)^2$.....(14)

Misalkan w_t adalah vektor bobot yang mengandung $w_j(t)$, b2(t), $v_{ij}(t)$ dan $b1_j(t)$. Gradien fungsi kinerja terhadap nilai bobot dan bias koneksi pada waktu ke-t didefinisikan dengan :

$g_{j(t)} = - \delta 2(t).\ z_j\ (t)$

$g_{b2(t)} = -\delta 2(t)$

$g_{ij(t)} = - \delta 1_j(t)$ xi

$g_{b1j(t)} = - \delta 1_j(t)$

g_t merupakan vektor gradien yang berisi $g_{j(t)}$, $g_{b2(t)}$, $g_{ij(t)}$ dan $g_{b1j(t)}$.

Jika $g_t = 0$ maka algoritma berhenti, jika tidak maka hitung : $\mathbf{d_t = - H_t.g_t}$ .....(15)

Langkah 3 : Tempatkan nilai α_t dengan menggunakan fungsi line search. Tujuannya adalah untuk meminimumkan *error* yang akan terjadi.

$\alpha_t = \arg\min \alpha[f(\mathbf{wt+\alpha tdt})]$

Perubahan vektor bobot dan bias yang terjadi adalah : $\mathbf{w_{t+1} = w_t + \alpha_t.\ d_t}$ .....(16)

Langkah 4 : Hitung perubahan bobot dan bias dengan persamaan berikut :

$\mathbf{\Delta\ wt = \alpha_t\ .dt}$ .....(17)

Sedangkan perubahan arah pencarian adalah sebagai berikut :

$\mathbf{\Delta\ gt = g_{t+1} - g_t}$ .....(18)

Maka didapatkan persamaan :

$$\left(1 \quad \frac{\qquad}{\qquad}\right).\frac{\qquad}{\qquad\qquad\qquad}$$ ..... (19)

Langkah 5 : *epoch*= *epoch*+ 1 Kembali ke langkah 2.

c. Algoritme *Lavemberg Marquardt*

Langkah dasar algoritme *Levenberg_Marquardt* adalah penentuan matriks *Hessian* untuk mencari bobot-bobot dan bias koneksi yang digunakan [9]. Matriks *Hessian* merupakan turunan kedua dari fungsi kinerja terhadap masing-masing komponen bobot dan bias. Untuk memudahkan proses komputasi, matriks *Hessian* diubah dengan pendekatan secara iteratif pada masing-masing *epoch* selama algoritma pelatihan berjalan. Proses perubahannya dilakukan dengan menggunakan fungsi gradien. Jika fungsi kinerja yang digunakan berbentuk jumlah kuadrat error (SSE), maka matriks *Hessian* dapat diestimasi dengan persamaan berikut:

$$\mathbf{H = J^T J+\eta I}$$ ….……(18)

dimana :

η : parameter *Marquardt*
I : matriks identitas
J : matriks Jakobian yang terdiri dari turunan pertama error jaringan terhadap masing-masing komponen bobot dan bias.

Matriks Jakobian dapat dikomputasikan melalui teknik B*ack propagation* standar [9]. Matriks Jakobian tersusun dari turunan pertama fungsi *error* terhadap masing-masing komponen bobot dan bias koneksi jaringan. Nilai parameter *Marquardt* (η) dapat berubah pada setiap *epoch*. Jika setelah berjalan satu *epoch* nilai fungsi *error* menjadi lebih kecil, nilai η akan dibagi oleh faktor τ. Bobot dan bias baru yang diperoleh akan dipertahankan dan pelatihan dapat dilanjutkan ke *epoch* berikutnya. Sebaliknya, jika setelah berjalan atau *epoch* nilai fungsi *error* menjadi lebih besar maka nilai η akan dikalikan dengan faktor τ. Nilai perubahan bobot dan bias dihitung kembali sehingga menghasilkan nilai yang baru.

Algoritme pelatihan dengan metode *Levenberg_Marquardt* dapat dijabarkan sebagai berikut :

Langkah 0 :

- Inisialisasi bobot awal dengan bilangan acak kecil
- Inisialisasi *Epoch* 0, MSE ≠0
- Tetapkan Maksimum *epoch*, parameter *Levenberg_Marquardt* (η > 0), faktor τ dan target Error

Langkah 1 :

Jika kondisi penghentian belum terpenuhi (*epoch*< Maksimum *epoch* atau MSE > target *Error*), lakukan langkah berikutnya.

Langkah 2 :

- *Epoch*= *epoch*+ 1
- Untuk setiap pasangan data pelatihan, lakukan langkah 3 – 4.

Langkah 3 :

Unit *output* Y menerima target pola yang berhubungan dengan pola input pelatihan. Jika diberikan N pasangan input data pelatihan (x_r, t_r), r = 1,2,...,N, dengan x_r adalah input dan target yang akan dicapai. Kesalahan pada suatu data pelatihan ke-r didefinisikan sebagai:

$\mathbf{e_r = t_r - y_r}$ ....(19)

dengan :

e_r : Kesalahan pada unit output

t_r : Keluaran yang diinginkan (acuan/ target)

y_r : Keluaran aktual.

e adalah vektor kesalahan berukuran Nx1 yang tersusun dari e_r , r = 1,2,...,N. e

dapat dituliskan sebagai :

$[e \quad]$ .....(20)

Misal bobot dan bias koneksi dinyatakan dalam vektor **w**, **w** merupakan vector berukuran ((2+n)p+1)x1 dapat dituliskan sebagai :

$[w \quad]$ .....(21)

Kesalahan suatu pelatihan jaringan oleh vektor bobot dan bias koneksi **w** pada suatu data pelatihan ke-r menjadi :

$(t \quad))$ .....(22)

Vektor kesalahan oleh vektor bobot dan bias koneksi **w** menjadi **e(w)** berukuran Nx1 yang tersusun dari e_r(**w**), dengan r = 1,2,...,N. Hitung fungsi jumlah kuadrat *error* dengan persamaan :

– .....(23)

Hitung matriks Jacobian untuk vektor bobot dan bias koneksi :

$[—]$ .....(24)

untuk r = 1,2,...,N

a. Hitung matriks Hessian untuk vektor bobot dan bias koneksi

$[J \quad nI]_{((2+n)p+1)x((2+n)p+1}$ .....(25)

b. Hitung perubahan vektor bobot dan bias dengan persamaan berikut :

$-[[H \quad)] \quad w]_{((2+n)p+1)x1}$.....(26)

c. Hitung vektor bobot dan bias baru.
w (baru) = w (lama) + Δw

d. Hitung kesalahan yang terjadi oleh bobot dan bias koneksi yang baru.

$E(w \quad))$

$-e(w \quad))T$ .....(27)

Bandingkan E(**w**) dengan E(**w(baru**)).

- Jika E(**w**) <= E(**w(baru**)) maka didapatkan η = η*τ dan kembali ke langkah a.
- Jika E(**w**) > E(**w(baru**)) maka didapatkan η = η/τ
  w (t+1)= **w** (t)+ Δ**w**

Kembali ke langkah 2.

5. Tahap kelima melakukan analisis dari hasil *training* masing-masing algoritme yang kemudian dibandingkan hasilnya.

III. HASIL DAN PEMBAHASAN

Forcasting Tuberculosis (TB) menggunakan pemodelan *neural network* dilakukan melalui beberapa iterasi, karakteristik utama yang akan dibahas dalam *neural network* untuk peramalan jumlah penderita TB adalah perancangan arsitektur jaringan, *training* dengan tiga algoritme yaitu *Back Propagation*, *Quasi Newton*, *Lavenberg Marquardt* yang diuji dan perbandingan hasil pengujian terhadap ketiga algoritma tersebut.

1. Perancangan Arsitektur ANN

Perancangan arsitektur jaringan ANN dilakukan dengan melakukan beberapa percobaaan untuk mendapatkan jaringan arsitektur terbaik. Percobaan perancangan arsitektur dengan 1 *hidden layer* dan 2 *hidden layer* dengan batasan 12 input, 8 *hidden layer* hidden 1 dan 5 *hidden layer* hidden 2 dengan jumlah *interasi* 500 dan diperoleh hasil dengan membandingkan nilai *fitness test error*, semakin nilai *fitness* lebih besar maka jaringan arsitekturnya semakin baik, untuk mendapatkan arsitektur terbaik maka dilakukan percobaan secara berulang-ulang dengan setiap percobaan mengkombinasikan arsitektur jaringan sebanyak 48 arsitektur. Hasil pencariaan dapat ditunjukkan pada table 1 berikut ini:

Tabel 1 Hasil Percobaan Pencarian Arsitektur Terbaik

No	Percobaan	Nilai Fittnes Test Error	Arsitektur Terbaik
1	Percobaan 1	0,024813	12-8-5-1
2	Percobaan 2	0,020729	12-8-3-1
3	Percobaan 3	0,023012	12-8-5-1
4	Percobaan 4	0,020474	12-8-4-1
5	Percobaan 5	0,018405	12-7-3-1
6	Percobaan 6	0,018808	12-7-5-1
7	Percobaan 7	0,019247	12-8-5-1
8	Percobaan 8	0,22927	12-8-5-1
9	Percobaan 9	0,020985	12-8-3-1
10	Percobaan 10	0,017967	12-6-3-1

Dari setiap percobaan diambil satu arsitektur terbaik dengan nilai *fittnes* terbaik kemudian diulang sebanyak 10 kali. Dari percobaan diperoleh sebuah arsitektur terbaik dengan jaringan 12-8-5-1 yang merupakan arsitektur yang dalam beberapa percobaan selalu yang terbaik dan yang memiliki nilai *fitness test error* terbesar adalah 0,024813. Berikut tabel hasil pengujian untuk mendapatkan arsitektur ANN terbaik.

2. Perbandingan Hasil Algoritme *Back Propogation, Quasi Newton* dan *Lavenberg Marquardt*

Setelah melakukan pengujian terhadap 3 macam algoritme yaitu algoritme *Backpropogation, Quasi Newton* dan *Lavenberg Marquardt* yang digunakan dalam prediksi jumlah penderita TB dengan melakukan *training* pada setiap algoritme dengan mengulang proses *training* masing-masing sebanyak tiga kali kemudian dari hasil *training* tersebut dibandingkan dan diambil nilai yang terbaik pada masing-masing algoritme untuk dibandingkan. Maka dapat dapat disimpulkan dari ketiga algoritme yang diuji dapat dilihat perbedaannya adalah sebagai berikut:

Tabel 2 Perbandingan hasil permalan tiga algoritma

	Back Propogation	*Quasi newton*	*Lavenberg Marquardt*
MAPE	11,9%	10,2 %	9,15 %
MSE	66719	53487	43419
MPE	-1,7%	-0,4%	-0,7%

a. *The Mean Squared Error* (MSE) adalah metode lain untuk mengevaluasi metode peramalan. Masing-masing kesalahan atau sisa dikuadratkan. Kemudian dijumlahkan dan dibagi dengan jumlah observasi. Pendekatan ini mengatur kesalahan peramalan yang besar karena kesalahan-kesalahan itu dikuadratkan. Suatu teknik yang menghasilkan kesalahan moderat mungkin lebih baik untuk salah satu yang memiliki kesalahan kecil tapi kadang-kadang menghasilkan sesuatu yang sangat besar. Dengan melihat hasil masing-masing nilai MSE maka algoritme *Lavenberg Marquardt* yang memiliki nilai paling kecil yaitu 43419 atau

dapat diketahui bahwa algoritme *Lavenberg Marquardt* pada peramalan ini memiliki kecendrungan kesalahan yang paling rendah dibanding dengan algoritme algoritme *Back Propogation* dan *Quasi newton.*

b. *The Mean Absolute Percentage Error* (MAPE) dihitung dengan menggunakan kesalahan *absolut* pada tiap periode dibagi dengan nilai observasi yang nyata untuk periode itu. Kemudian meratarata kesalahan persentase *absolut* tersebut. Pendekatan ini berguna ketika ukuran atau besar variabel ramalan itu penting dalam mengevaluasi ketepatan ramalan. MAPE mengindikasi seberapa besar kesalahan dalam meramal yang dibandingkan dengan nilai nyata pada deret. Metode MAPE digunakan jika nilai Yt besar. MAPE juga dapat digunakan untuk membandingkan ketepatan dari teknik yang sama atau berbeda dalam dua deret yang sangat berbeda dan mengukur ketepatan nilai dugaan model yang dinyatakan dalam bentuk rata-rata persentase *absolut* kesalahan. Nilai MAPE yang paling mendekati nol adalah nilai MAPE pada algoritme *Lavenberg Marquardt.*

c. Dari hasil training terhadap ketiga algoritme dapat diukur dengan membandingkan nilai *MAPE* dan MSE dari tiap-tiap algoritme. MSE adalah metode lain untuk mengevaluasi metode peramalan. Sedangkan MAPE ini berguna ketika ukuran atau besar variabel ramalan itu penting dalam mengevaluasi ketepatan ramalan. MAPE mengindikasi seberapa besar kesalahan dalam meramal yang dibandingkan dengan nilai nyata pada deret. Dari ketiga algoritme dapat dilihat bahwa nilai *MAPE* dan MSE dengan nilai terendah dan merupakan yang terbaik adalah pada algoritme *lavenberg marquardt*.

d. Nilai MPE adalah merupakan metode untuk menentukan apakah suatu metode peramalan bias (peramalan tinggi atau rendah secara konsisten). MPE dihitung dengan mencari kesalahan pada tiap periode dibagi dengan nilai nyata untuk periode itu. Kemudian, merata-rata kesalahan persentase ini. Jika pendekatan peramalan tak bias, MPE akan menghasilkan angka yang mendekati nol. Dengan demikian dari ketiga algoritma nilai MPE paling kecil dimiliki algoritme *Quasi Newton* yaitu 0,4% mengindikasikan bahwa teknik ini tidak bias. Karena hasilnya mendekati nol, teknik ini tidak selamanya konsisten atau mengabaikan jumlah korban TB tiap bulannya.

e. Dalam hasil testing terhadap ketiga algoritme menunjukkan bahwa nilai target dengan nilai *output* pada masing-masing algoritme terdapat perbedaan, semakin dekat nilai target dengan nilai output maka semakin baik. Dari ketiga algoritma yang dibandingkan bahwa algoritme *quasi newton* memiliki nilai output yang lebih dekat dengan dengan nilai target yaitu dengan nilai target sebesar 1357,66 diperoleh *mean forecasting* sebesar 1356,06. Dengan demikian algoritme *Quasi Newton* yang terbaik dalam melakukan *training* terhadap nilai aktual dengan nilai *output*.

IV. KESIMPULAN

Hasil pengujian dengan membandingan tiga algoritma menyatakan bahwa algoritma *Lavenberg Marquardt* lebih baik dibanding algoritme *Back Propagation*, dan *Quasi Newton* dalam melakukan peramalan dengan menggunakan ANN.

REFERENSI

[1] N. Ganesan, K. Venkatesh, and M. A. Rama (2010), "*Application of Neural Networks in Diagnosing Cancer Disease Using Demographic Data*", *International Journal of Computer Applications (0975 - 8887).*

[2] Freeman, A. James (2015) "*Network with Back Propagation, International Conference on Computing and Intelligence Systems"* Volume: 04, Special Issue:March 2015 Pages: 1166 – 1169

[3] N.Guru, A. Dahiya, N. Rajpal (2007), " *Decision support system for Heart Disease Diagnosis Using Neural Network*", Delhi Business Review, vol. 8, No.1,Jan-June.

[4] G.M.Nasiraand, S.Radhimeenakshi (2014), " *A study on prediction of Cardiovascular Victimization Data Processing Techniques", International Journal of computer and Organization Trends, vol 9. Number 1, pp. 32- 35.*

[5] J. Shi, M. Chui (2012), " *Extract Knowledge from Site-sampled Data sets and Fused Hierarchical Neural Networks for detecting Cardiovascular Diseases" , International conference on Biomedical Engineering and Biotechnology.*

[6] J.J. Siang, (2005), " Jaringan Syaraf Tiruan & Pemprograman Menggunakan Matlab", Penerbit Andi, Yogyakarta.

[7] Makridakis, S.Wheelwright (1999)." Metode dan Aplikasi Peramalan Jilid 1" (Ir. Untung sus Ardiyanto, M.Sc. & Ir. Abdul Basith, M.Sc Terjemah). Edisi Kedua, Jakarta: Penerbit Erlangga.

[8] S. Afshar, F. Abdolrahmani (2011), "*Recognition and prediction of leukemia with Artificial Neural Network (ANN)", Medical Journal of Islamic Republic of Iran, Vol. 25, No. 1, pp. 35-39.*

[9] M.S. Al-Haik, H. Garmestani and I.M. Navon (2003), "*Truncated-Newton Training Algorithm for Neurocomputational Viscoplastic Model", Comput. Methods Appl*. Mech. Engrg., No. 192, p. 2249-2.