SEMINAR NASIONAL TEKNOKA

Vol. 5, 2020
ISSN No. **2502-8782**

Prediksi Jumlah Penderita COVID-19 di Kota Malang Menggunakan Jaringan Syaraf Tiruan *Backpropagation* dan Metode *Conjugate Gradient*

Syaiful Anam[1)], Mochamad Hakim Akbar Assidiq Maulana[2)], Noor Hidayat [3)], Indah Yanti [4)], Zuraidah Fitriah[5)] & Dwi Mifta Mahanani[6)]

[1,2,3,4,5,6)]Kelompok Kajian Matematika Optimasi dan Komputasi, Universitas Brawijaya, Jl. Veteran, Kota Malang
[1,2,3,4,5,6)]Jurusan Matematika, Universitas Brawijaya, Jl. Veteran Malang Telp 0341 571142
E-mail: syaiful@ub.ac.id

Abstrak-*COVID-19 merupakan penyakit menular yang diakibatkan oleh infeksi virus corona baru. Penyakit ini sangat berbahaya dan menyebabkan kematian terutama bagi penderita yang memilki penyakit bawaan atau yang memiliki imunitas rendah. Penyebaran penyakit ini melalui melalui percikan-percikan dari hidung atau mulut yang keluar saat orang yang terinfeksi COVID-19 batuk, bersin atau berbicara. Prediksi jumlah penderita COVID-19 menjadi sangat penting untuk dilakukan dalam pencegahan dan penanggulangan penyebaran penyakit ini. Jaringan syaraf tiruan backpropagation merupakan salah metode yang dapat digunakan untuk menyelesaikan masalah prediksi dengan hasil yang baik, tetapi kinerjanya dipengaruhi oleh metode optimisasi yang digunakan saat pelatihan. Pada umumnya metode optimisasi yang digunakan adalah metode gradient descent, tetapi metode ini memiliki konvergensi yang lambat. Metode Conjugate Gradient memiliki konvergensi yang sangat baik jika dibandingkan dengan metode gradient descent. Pada tulisan ini akan dibahas bagaimana membuat model prediksi jumlah penderita COVID-19 di Kota Malang menggunakan jaringan syaraf backprogation dan metode conjugate gradient. Hasil eksperimen menunjukan bahwa model prediksi ini memperoleh hasil yang baik jika dibandingkan jaringan syaraf tiruan yang dioptimasi dengan metode gradient descent.*

Kata kunci: *backpropagation, conjugate gradient, jaringan syaraf tiruan, penderita COVID-19, prediksi*

Abstract-*COVID-19 is an infectious disease caused by infection with a new type of corona virus. This disease is very dangerous and causes death, especially for sufferers who have congenital diseases or who have low immunity. The disease is spread through droplets from the nose or mouth that come out when a person infected with COVID-19 coughs, sneezes or talks. The prediction of the number of COVID-19 sufferers is very important to prevent and combat the spread of this disease. The backpropagation neural network is a method that can be used to solve predictive problems with good results, but its performance is influenced by the optimization method used during training. In general, the optimization method used is the gradient descent method, but this method has slow convergence. The Conjugate Gradient method has very good convergence when compared to the gradient descent method. In this paper, we will discuss how to make a prediction model for the number of COVID-19 sufferers in Malang using the backpropagation neural network and the conjugate gradient method. The experimental results show that the prediction model gets good results when compared to artificial neural networks that are optimized by the gradient descent method.*

Keywords: *artificial neural networks, backpropagation, conjugate gradient, predictions, sufferers of COVID-19*

1 PENDAHULUAN

Pada akhir bulan Desember 2019, Indonesia dan dunia digemparkan dengan kemunculan sebuah penyakit menular yang menyerang organ pernapasan. Penyakit ini oleh WHO (*World Health Organization*) diberi nama COVID-19 [1]. Penyakit COVID-19 diakibatkan oleh infeksi dari jenis *corona virus* tipe baru. Virus ini pertama kali ditemukan di Kota Wuhan, Provinsi Hubei, China kemudian menyebar ke seluruh penjuru dunia termasuk Indonesia melalui kontak langsung dengan penderita penyakit dari kota asalnya [2].

Akibat dari penyakit ini sangat serius karena pernapasan termasuk organ vital manusia yang membantu proses metabolisme dan kesetimbangan zat-zat dalam tubuh. Selain itu, penyakit ini juga

Dipresentasikan pada Tanggal 28 November 2020
DOI: 10.22236/teknoka.v5i.367

berdampak fatal yaitu menimbulkan kematian bagi penderitanya [3], terlebih lagi jika pasien tersebut juga mempunyai penyakit bawaan ataupun imunitas pasien yang rendah. Penyakit ini mudah menyebar karena seperti penyakit pernapasan menular lainnya.

Penularan terjadi karena percikan-percikan droplet yang keluar dari hidung atau mulut saat orang yang telah terinfeksi covid19 tersebut batuk, bersin atau berbicara. Oleh karena itu pada masa pandemik sangat dianjurkan untuk menggunakan masker ataupun alat pelindung serta melakukan pembatasan sosial untuk mengurangi potensi penyebaran virus [4].

Jumlah penderita COVID-19 mengalami kenaikan setiap harinya. Peningkatan jumlah pasien penyakit ini sebaiknya berbanding lurus dengan pelayanan kesehatan yang memadai sehingga tingkat laju penyebaran virus dapat dikurangi. Prediksi jumlah penderita COVID-19 berdasarkan data dari jumlah penderita yang telah ada sebelumnya sangat diperlukan untuk pencegahan penyakit ini dan penyediaan fasilitas pelayanan kesehatan di masa yang akan datang [5].

Pertambahan jumlah penderita COVID-19 dipengaruhi oleh beberapa faktor dari penyebaran virus, misalnya jumlah kematian penderita dan kasus sembuhnya pasien dari penyakit ini. Selain itu, masa inkubasi virus di dalam tubuh manusia yaitu selama 14 hari juga mempengaruhi perkiraan dari jumlah data pada hari setelahnya [6].

Banyak metode telah diusulkan untuk memprediksi penyebaran suatu virus. Virus dapat dimodelkan sebagai populasi yang dipengaruhi oleh faktor persebaran penyakit itu sendiri pada manusia. Salah satu jenis prediksi adalah analisis deret waktu yaitu mencari variabel dengan variabel yang mempengaruhinya dan dikaitkan dengan waktu ataupun dengan menganalisis sebab akibatnya saja. Jenis prediksi penyebaran penyakit ini mengarah ke analisis deret waktu dikarenakan jumlah penderita COVID-19 saat ini dipengaruhi jumlah penderita waktu sebelumnya. Metode regresi biasanya digunakan dalam permasalahan deret waktu. Terdapat dua jenis regresi yaitu regresi linear dan regresi non linear. Pertumbuhan suatu populasi dianggap kurang realistis dengan keadaan yang ada di lingkungan, sehingga regresi non linear digunakan untuk mengatasi jumlah kesalahan prediksi yang diinginkan. Walaupun begitu regresi non linear kurang baik jika bekerja pada faktor yang lebih kompleks. Jaringan Syaraf Tiruan (JST) adalah alternatif dari metode prediksi. JST jauh lebih fleksibel dan dapat menangani kasus yang lebih rumit dan tak berasumsi dibanding dengan metode regresi.

JST mempunyai beberapa jenis berdasarkan algoritma prediksi yang digunakan, salah satunya adalah JST *Backpropagation*. JST *Backpropagation* merupakan metode yang dapat digunakan untuk menyelesaikan masalah prediksi dengan hasil yang baik, tetapi kinerjanya dipengaruhi oleh metode optimisasi yang digunakan saat pelatihan. Pada umumnya metode optimisasi yang digunakan adalah metode *gradient descent*, tetapi metode ini memiliki konvergensi yang lambat [7]. Metode optimasi lain dapat mengatasi masalah kekonvergenan tersebut adalah metode *Conjugate Gradient*. Metode yang satu ini memiliki konvergensi yang sangat baik jika dibandingkan dengan metode *Gradient Descent* [8].

Pada tulisan ini akan dibahas bagaimana membuat model prediksi jumlah penderita COVID-19 di Kota Malang menggunakan jaringan syaraf *backpropagation* dan metode optimasi *Conjugate Gradient*. Eksperimen ini membandingkan JST yang dioptimasi *Conjugate Gradient* dengan JST yang dioptimasi metode *Gradient Descent*. Model prediksi didapatkan melalui serangkaian percobaan dengan mengkombinasikan arsitektur jaringan dan *learning rate* untuk mendapatkan model prediksi yang paling optimum.

2 LANDASAN TEORI

Pada bagian ini akan dibahas beberapa teori yang berkaitan dengan penelitian ini, antara lain definisi optimisasi, metode *Conjugate Gradient*, jaringan syaraf tiruan (JST) dan JST *backpropagation*. JST dapat dianalogi sebagai fungsi yang digunakan untuk memprediksi sesuatu, dalam penelitian ini jaringan syaraf tiruan digunakan untuk memprediksi jumlah penderita COVID-19.

2.1 Optimisasi

Optimisasi adalah suatu proses mencari solusi terbaik atau nilai optimal dari permasalahan. Metode optimisasi tersebut digunakan untuk mencari nilai maksimal atau nilai minimal. Permasalahan optimisasi banyak dijumpai dalam kehidupan sehari-hari, seperti bidang matematika, teknik, sosial, ekonomi, pertanian, farmasi, otomotif, dan lain-lain [9]. Fungsi tujuan dapat dicari nilai optimalnya pada suatu titik menggunakan metode-metode optimisasi. Berbagai macam metode dari optimisasi telah diciptakan seperti *Golden Search* dan *Quadratic Approximation* untuk masalah sederhana fungsi objektif satu dimensi serta *Gradient Descent*, *Conjugate Gradient*, *Newton*, dan lain-lain.

2.2 Metode *Conjugate Gradient*

Metode Newton memiliki sifat yang disebut terminasi kuadrat, sehingga dapat meminimalkan fungsi kuadrat dengan tepat dalam iterasi yang terbatas, tetapi membutuhkan penghitungan dan penyimpanan turunan kedua dari fungsi. Jika jumlah parameter sangat besar, maka metode Newton menjadi tidak praktis untuk menghitung semua turunan keduanya.
JST memerlukan beberapa ratus hingga ribuan bobot, sehingga penggunaan metode optimisasi yang membutuhkan perhitungan turunan kurang praktis.

DOI: 10.22236/teknoka.v5i.367
ISSN No. 2502-8782

Oleh karena itu, JST membutuhkan metode optimisasi yang hanya memerlukan turunan pertama dan memiliki penghentian kuadrat [10].

Metode optimisasi yang lainnya adalah metode Conjugate Gradient. Metode Conjugate Gradient merupakan salah satu metode iteratif untuk menyelesaikan sistem persamaan linear. Metode ini efektif untuk sistem persamaan linear dengan matriks koefisien yang simetrik definit positif. Secara umum, metode ini membangkitkan vektor-vektor yang *conjugate* dan juga merupakan *gradient* dari fungsi kuadrat. Metode ini menyelesaikan sistem persamaan linear dengan cara menemukan titik minimum dari fungsi kuadrat. Berikut ini adalah algoritma dari metode *Conjugate Gradient.*

Algoritma *Conjugate Gradient*:

a. Memasukkan $\mathbf{x}_0$, ε (kriteria berhenti), dan k_{maks}
b. Menginisialisasi iterasi k=0
c. Menghitung nilai $\boldsymbol{p_0}$ dengan persamaan (1),
$$\boldsymbol{p_0} = -\boldsymbol{g_0} = \nabla f(\boldsymbol{x_0}) \quad (1)$$
yang merupakan arah pencarian pertama yaitu negatif dari gradient fungsi
d. While $(\|\nabla f(\boldsymbol{x_k})\| > \varepsilon)$ or (k<= k_{maks}) do
   1. Menghitung nilai $\boldsymbol{p_k}$ dengan persamaan (2)
   $$\boldsymbol{p_k} = -\boldsymbol{g_k} + \beta_k \boldsymbol{p_{k-1}} \quad (2)$$
   2. Pilih $\propto$ yang meminimumkan $f\ (\mathbf{x_k} + \alpha \boldsymbol{p_k})$
   3. Menghitung nilai $\mathbf{x_{k+1}}$ dengan persamaan (3),
   $$\mathbf{x_{k+1}} = \mathbf{x_k} + \alpha \boldsymbol{p_k} \quad (3)$$
   dimana α merupakan koefisien pembelajaran / *learning rate*
   4. Menghitung nilai β_k yang didapat dari persamaan (4),
   $$\beta_k = \frac{\boldsymbol{g_k}^T \boldsymbol{g_k}}{\boldsymbol{g_{k-1}}^T \boldsymbol{g_{k-1}}} \quad (4)$$
   dimana merupakan metode *Fletcher and Reeves*
e. End While

2.3 Jaringan Syaraf Tiruan

JST merupakan suatu sistem pemrosesan informasi yang memilki karakteristik menyerupai jaringan syaraf biologi. JST teranalogi dari generalisasi model matematis pemahaman manusia (*human cognition*). Jaringan syaraf terdiri dari beberapa *neuron*, dan terdapat hubungan antara *neuron*. *Neuron* akan mentransformasikan informasi yang diterima menuju ke *neuron-neuron* yang lain. Hubungan pada JST ini kemudian dikenal dengan sebutan bobot [11].

JST dikarakterisasi oleh pola hubungan antar setiap neuron (kemudian disebut dengan arsitektur), metode untuk menentukan bobot di setiap hubungan antar neuron (disebut *training* atau algoritma pembelajaran), dan fungsi aktivasi [12].

2.4 Jaringan Syaraf Tiruan *Backpropagation*

Algoritma *Backpropagation* adalah sebuah metode sistematik untuk melakukan pelatihan pada *layer* JST. JST *Backpropagation* merupakan salah satu algoritma yang sering digunakan dalam menyelesaikan masalah-masalah yang rumit. JST *Backpropagation* terdiri dari *n* buah masukan ditambah sebuah bias, sebuah layar tersembunyi yang terdiri dari *p* unit dan sebuah bias, serta *m* buah unit keluaran. Bias V_{0j} dan W_{0k} berperilaku seperti bobot dimana *output* bias ini selalu sama dengan 1 [11]. Proses *training* pada JST *backpropagation* mempunyai tiga tahapan yaitu langkah maju (*feedforward*) dari training masukan pola, langkah mundur (*backpropagation*) dari *error* yang terasosiasi, dan penyesuaian (*updating)* bobot. Selama langkah maju setiap unit input akan dihitung didalam *layer-layer* tersembunyi sehingga mendapatkan *output* dari pola. Selama proses *training* hasil *output* dari jaringan akan dibandingkan dengan target kemudian dihitung *error* dan dilakukan optimisasi untuk selanjutnya didapatkan faktor yang mendidtribusikan *error*. Faktor tersebut digunakan untuk meng-*update* bobot yang berada di antara *layer input* dan *layer output* [12].

Algoritma *Backpropagation*:

a. Menginisialisasi bobot
b. Kerjakan selama kondisi berhenti bernilai salah
   1. *Feedforward*
      - Setiap input unit ($\boldsymbol{X}_i$, i=1, 2, 3, ... *n*) menerima sinyal masukan dan disebarkan ke *layer* tersembunyi setelahnya
      - Setiap unit tersembunyi (z_j, *j*=1, 2, 3, ... *m*) menjumlahkan *input* yang telah diberi bobot seperti terlihat pada persamaan (5).
      $$z\_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij} \quad (5)$$
      Selanjutnya sinyal *output* dihitung menggunakan fungsi aktifasi, kemudian nilainya disebarkan kembali ke *layer* setelahnya dengan menggunakan persamaan (6).
      $$z_j = f(z\_in_j) \quad (6)$$
      - Setiap output unit (y_k, *k*=1, 2, 3, ... *p*) menjumlahkan input yang telah diberi bobot dengan menggunakan persamaan (7).
      $$y\_in_k = w_{0k} + \sum_{j=1}^{m} z_j w_{jk} \quad (7)$$
      Sinyal output dihitung dengan menggunakan fungsi aktivasi pada persamaan (8).
      $$y_k = f(y\_in_k) \quad (8)$$
   2. *Backpropagation* dari *error*
      - Setiap *output* unit (y_k, *k*=1, 2, 3, ... *p*) mendapatkan pola yang berkorespondensi dengan pola *input training*, selanjutnya menghitung informasi *error* dengan persamaan (9).
      $$\delta_k = (t_k - y_k) f'(y\_in_k) \quad (9)$$

DOI: 10.22236/teknoka.v5i.367
ISSN No. 2502-8782

- Selanjutnya, menghitung koreksi *error* untuk meng-*update* bobot nantinya dengan persamaan (10).
  - $\Delta w_{jk} = \alpha \delta_k z_j$ (10)

  Koreksi bias juga dihitung dengan persamaan (11), kemudian menyampaikan δ_k ke unit-unit *layer* setelahnya

  $$\Delta w_{0k} = \alpha \delta_k \quad (11)$$
- Setiap unit tersembunyi (z_j, j=1, 2, 3, ... *m*) jumlahkan dengan delta input (dari unit pada *layer* sebelumnya) yang dihitung dengan persamaan (12).

  $$\delta\_in_j = \sum_{k=1}^{p} \delta_k w_{jk} \quad (12)$$

  Kalikan dengan turunan dari fungsi aktivasi untuk menghitung informasi *error* dengan persamaan (13).

  $$\delta_j = \delta\_in_j f'(z\_in_j) \quad (13)$$

  Selanjutnya, menghitung koreksi *error* dengan persamaan (14) untuk meng-*update* bobot nantinya.

  $$\Delta v_{ij} = \alpha \delta_j x_i \quad (14)$$

  Selanjutnya, koreksi bias dihitung dengan persamaan (15).

  $$\Delta v_{0j} = \alpha \delta_j \quad (15)$$

3. Meng-*update* bobot dan bias
   - Setiap output unit (y_k, *k*=1, 2, 3, ... *p*) di-*update* bias dan bobot (j=0,…m):

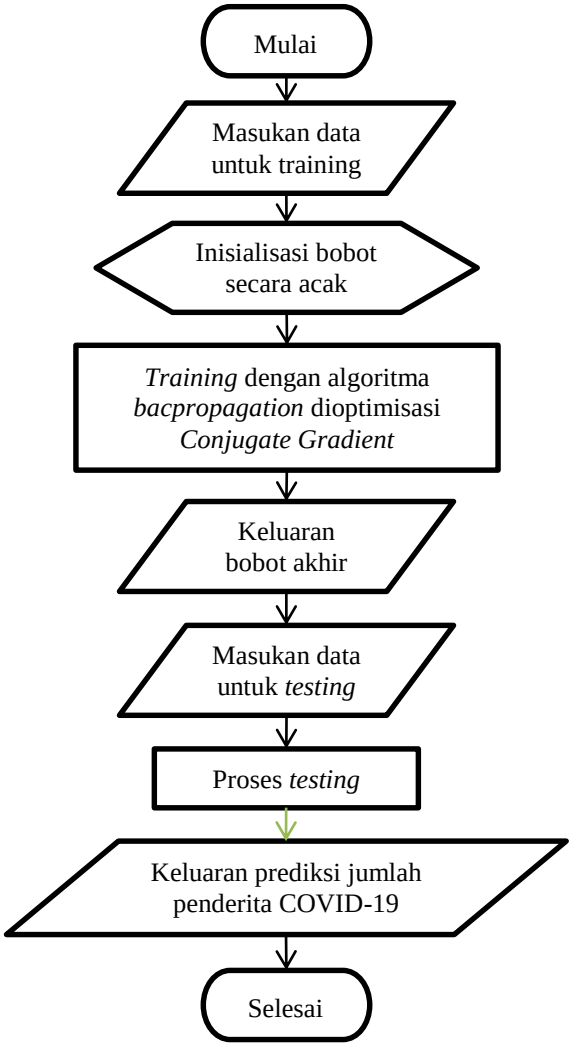
     $$w_{jk}{}^{baru} = w_{jk}{}^{lama} + \Delta w_{jk} \quad (16)$$
   - Setiap unit tersembunyi (z_j, *j*=1, 2, 3, ... *m*) *update* bias dan bobot (i=0,…n):

     $$v_{ij}{}^{baru} = v_{ij}{}^{lama} + \Delta v_{ij} \quad (17)$$

c. Cek kondisi berhenti

3 METODOLOGI PENELITIAN

Diagram dari metode/sistem yang digunakan untuk memprediksi peningkatan jumlah penderita COVID-19 ditunjukkan pada Gambar 1. Dengan menggunakan sistem ini, peningkatan jumlah penderita COVID-19 di Kota Malang dapat diprediksi berdasarkan faktor-faktor dari penyebaran virus COVID-19.



***Gambar 1** Diagram dari sistem untuk memprediksi jumlah penderita COVID-19 dengan JST backpropagation yang dioptimisasi oleh metode Conjugate Gradient.*

Langkah pertama adalah melakukan preproses data dengan menormalisasikan data ke dengan mentransformasikannya dalam jangkauan 0 dan 1. Hal ini dapat dilakukan dengan cara membagi seluruh data yang ada dengan jumlah populasi pada suatu tempat (dalam hal ini yaitu jumlah populasi di Kota Malang). Akan tetapi jumlah penderita COVID-19 masih terlampau sangat kecil sehingga pola keluaran yang didapat tidak maksimal, sehingga pembagi untuk normalisasi hanya sebesar 25% dari populasi Kota Malang.

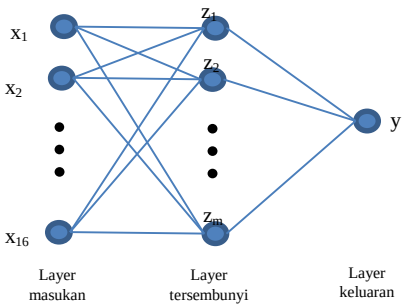
Selanjutnya yang dilakukan adalah menentukan *input* dari data *training* dari faktor-faktor dari penyebaran virus COVID-19 seperti jumlah kematian penderita, kasus sembuhnya pasien dari penyakit, dan data peningkatan jumlah pasien14 hari sebelum hari yang diprediksi di-*input*-kan kedalam sistem.

DOI: 10.22236/teknoka.v5i.367

ISSN No. 2502-8782

Inisialisasi bobot dari JST *Backpropagation* dibangkitkan secara acak. Langkah selanjutnya JST *Backpropagation* melakukan *training berdasarkan data training*. Pada langkah ini, bobot dari jaringan akan diperbaiki dengan meminimalkan *error* antara *output* dari jaringan dan target. *Error* tersebut diminimalkan dengan metode *Conjugate Gradient*. Bobot akhir digunakan untuk langkah pengujian jaringan. Pada tahap pengujian, data *testing* akan di-*input* dan hasil *output*-nya berupa prediksi jumlah penderita COVID-19.

Hipotesis dari penelitian ini adalah jumlah kematian penderita dan kasus sembuhnya pasien dari penyakit serta data peningkatan pasien 14 hari sebelumnya akan mempengaruhi peningkatan jumlah penderita COVID-19 hari ini. *Input* data program / sistem ini adalah jumlah kematian penderita dan kasus sembuhnya pasien dari penyakit serta data peningkatan jumlah pasien COVID-19 pada setiap 14 hari sebelumnya.



***Gambar 2** Arsitektur JST backpropagation.*

Arsitektur jaringan dibangun berdasarkan banyaknya faktor yang mempengaruhi peningkatan jumlah pasien COVID-19. Faktor-faktor tersebut antara lain jumlah kematian penderita dan kasus sembuhnya pasien dari penyakit, data peningkatan jumlah pasien COVID-19 pada setiap 14 hari sebelumnya. Variabel yang digunakan sebagai input jaringan adalah 16 variabel adalah data peningkatan jumlah pasien COVID-19 pada setiap 14 hari sebelumnya (x_1, x_2, x_3, … x_{14}), jumlah kematian penderita (x_{15}) dan kasus sembuhnya pasien dari penyakit (x_{16}), sedangkan variabel yang digunakan sebagai *output* jaringan adalah peningkatan jumlah pasien COVID-19 (y). Arsitektur jaringan ditunjukkan pada Gambar 2. Pada Gambar 2 tersebut arsitektur jaringan memiliki tiga *layer* yaitu *layer* masukan, *layer* tersembunyi dan *layer* keluaran.

Jumlah *neuron* tersembunyi dalam jaringan ditentukan dengan uji coba (*trial and error*). Tujuannya adalah untuk mengetahui arsitektur paling baik untuk mengenali pola secara akurat. Oleh karena itu, dalam penelitian ini diambil model arsitektur yang meliputi: 16-5-1, 16-20-1, 16-50-1, 16-100-1, dan 16-150-1. Maksud dari model arsitektur 16-5-1 adalah bahwa jaringan dibangun oleh 16 neuron pada *layer* masukan dan 1 neuron pada *layer* keluaran sedangkan jumlah *neuron* pada *layer* tersembunyi adalah sebanyak 5 *neuron*.

Pelatihan jaringan dengan algoritma *Backpropagation* memiliki tiga tahap yaitu *feedforward* dari data input, *Backpropagation* dari kesalahan, dan proses *update* bobot.

Algoritma *Backpropagation* dengan *Conjugate Gradient*:

a. Inisialisasi it=1 serta bobot, v_{ij} dan w_{jk}
b. Menentukan itmaks
c. While it<itmaks

1. *Feedforward*
   - Setiap input unit (X_i, i=1, 2, 3, ... 16) menerima sinyal masukan dan disebarkan ke *layer* tersembunyi setelahnya
   - Setiap unit tersembunyi (z_j, j=1, 2, ... m, I menurut percobaan) menjumlahkan *input* yang telah diberi bobot dengan persamaan
   $$z\_in_j = v_{0j} + \sum_{i=1}^{16} x_i v_{ij} \quad (17)$$
   Menggunakan fungsi aktifasi untuk menghitung sinyal output, kemudian nilainya disebarkan kembali ke *layer* setelahnya. Fungsi aktivasi yang digunakan untuk *layer* tersembunyi dan *layer output* adalah fungsi *logsigmoid*
   $$z_j = f(z\_in_j) = \frac{1}{1+e^{-z\_in_j}} \quad (18)$$
   - *Output unit* (y) menjumlahkan *input* yang telah diberi bobot
   $$y\_in = w_0 + \sum_{j=1}^{m} z_j w_j \quad (19)$$
   Menggunakan fungsi aktivasi untuk menghitung sinyal *output*
   $$y = f(y\_in) = \frac{1}{1+e^{-y\_in}} \quad (20)$$
2. *Backpropagation* dari *error*
   - *Output unit* (y) mendapatkan pola yang berkorespondensi dengan pola *input training*, hitung informasi *error*
   $$\delta = (t-t)f'(y\_in) \quad (21)$$
   - Setiap unit tersembunyi (Z_j, j=1, 2, 3, ... m) jumlahkan dengan delta input (dari unit pada *layer* sebelumnya)
   $$\delta\_in_j = \delta w_j \quad (22)$$
3. Meng-*update* bobot dan bias
   - Hitung nilai $\boldsymbol{p_{it}}$
   $$\boldsymbol{p_{it}} = -\boldsymbol{g_{it}} + \beta_{it}\boldsymbol{p_{it-1}} \quad (23)$$
   Nilai β_{it} didapat dari
   $$\beta_{it} = \frac{\boldsymbol{g_{it}}^T \boldsymbol{g_{it}}}{\boldsymbol{g_{it-1}}^T \boldsymbol{g_{it-1}}} \quad (24)$$

DOI: 10.22236/teknoka.v5i.367

- *Output unit* (y) di-*update* bias dan bobot (j=0,…m):

$$w_j^{it+1} = w_j^{it} + \alpha \boldsymbol{p_{it}}(\delta z_j) \quad (25)$$

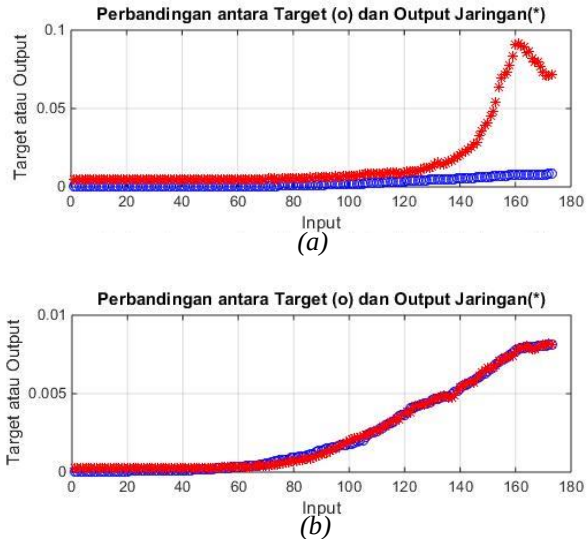
- Setiap unit tersembunyi (Z_j, j=1, 2, 3, ... m) *update* bias dan bobot (i=0,…16):

$$v_{ij}^{it+1} = v_{ij}^{it} + \alpha \boldsymbol{p_{it}}(\delta_j x_i) \quad (26)$$

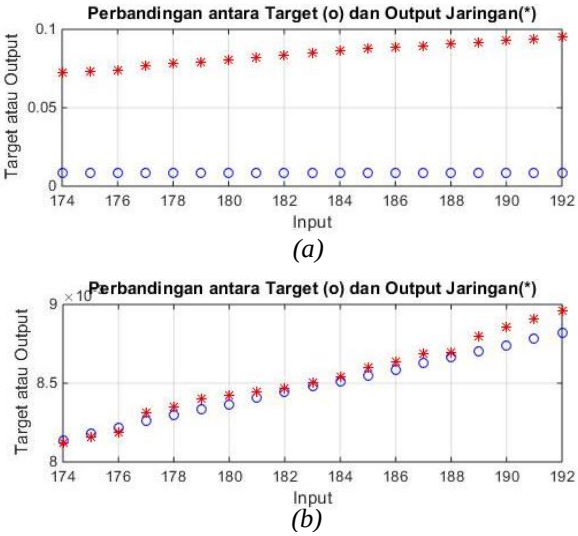
d. Cek kondisi berhenti

4 HASIL DAN PEMBAHASAN

Evaluasi dari prediksi jumlah pasien COVID-19 digunakan data yang diambil dari website GUGUS TUGAS COVID-19 KOTA MALANG yang juga dipublikasikan pada akun sosial media instagram @pemkotmalang. Banyak data yang diambil yaitu sebanyak 192 data yang diambil dari kasus pada bulan Maret sampai Oktober 2020. Data yang telah diperoleh dibagi menjadi dua yaitu data *training* dan data *testing*. Banyaknya data yang digunakan untuk *training* yaitu sebanyak 173 data (90% dari total data) dan banyak data untuk *testing* yaitu 19 data (10% dari total data. Untuk mengimplementasikan sistem prediksi, perangkat keras yang digunakan yaitu laptop dengan prosesor *Core i3 7th Gen*, 2.30 GHz, 8192MB RAM, 250 SSD dan perangkat lunak yang digunakan adalah MATLAB R2014b.



Gambar 3 *Perbandingan antara target dan keluaran jaringan untuk data training. (a). JST Backpropagation yang dioptimisasi dengan metode Gradient Descent (b) JST Backpropagation yang dioptimisasi dengan metode Conjugate Gradient*



Gambar 4 *Perbandingan antara target dan keluaran jaringan untuk data testing. (a). JST Backpropagation yang dioptimisasi dengan metode Gradient Descent (b) JST Backpropagation yang dioptimisasi dengan metode Conjugate Gradient*

Input dari JST adalah jumlah kasus yang ada untuk setiap 14 hari sebelumnya, jumlah kasus pasien yang meninggal, dan jumlah kasus pasien yang sembuh dari COVID-19 sampai hari sebelumnya serta *output* dari jaringan tersebut adalah banyaknya kasus yang terkonfirmasi sampai hari ini. Tabel 1 dan 2 menunjukkan perbandingan performa dari metode *Conjugate Gradient* dan *Gradient Descent* untuk mengoptimisasi JST *Backpropagation* dengan beberapa arsitektur yang berbeda. Tabel-tabel tersebut memperlihatkan bahwa metode *Conjugate Gradient* memiliki akurasi yang lebih baik dari pada *Gradient Descent*. Metode *Conjugate Gradient* dan *Gradient Descent* memiliki waktu komputasi yang hampir sama untuk arsitektur JST dengan jumlah *neuron* tersembunyi yang sedikit. Banyaknya *layer* tersembunyi tidak boleh terlalu sedikit atau terlalu banyak. Jumlah *neuron* tersembunyi yang terlalu banyak menyebabkan *overfitting*, artinya bahwa akurasi dari data *training* yang sangat baik tetapi akurasi dari data *testing* masih terlalu jelek. Tabel 3 dan 4 menunjukkan perbandingan metode *Conjugate Gradient* dan *Gradient Descent* untuk memprediksi jumlah kasus terkonfirmasi penderita COVID-19. Tabel-tabel tersebut memperlihatkan bahwa laju pembelajaran terbaiknya yaitu 0.01.

Tabel 1 *Hasil evaluasi dengan menggunakan data training untuk beberapa arsitektur dengan learning rate 0.005.*

DOI: 10.22236/teknoka.v5i.367

ISSN No. 2502-8782

Arsitektur	*MSE (Means Square Error)*		*Durasi Komputasi*	
	Gradient Descent	*Conjugate Gradient*	*Gradient Descent*	*Conjugate Gradient*
16-5-1	14471.31	**1.31**	2.03	**0.80**
16-20-1	1458.48	**1.44**	2.13	**1.09**
16-50-1	394.77	**1.27**	2.46	**2.01**
16-100-1	168.11	**1.30**	**4.37**	5.09
16-150-1	105.45	**1.97**	5.37	**3.40**

Tabel 2 *Hasil evaluasi dengan menggunakan data testing untuk beberapa arsitektur dengan learning rate 0.005.*

Arsitektur	*MSE (Means Square Error)*	
	Gradient Descent	*Conjugate Gradient*
16-5-1	*32391.95*	***6.67***
16-20-1	*11092.24*	***11.49***
16-50-1	*2901.01*	***27.18***
16-100-1	*1259.15*	***881.52***
16-150-1	*915.89*	***595.43***

Tabel 3 *Hasil evaluasi dengan menggunakan data training untuk beberapa learning rate dengan 50 neuron tersembunyi.*

Learning Rate	*MSE (Means Square Error)*		*Durasi Komputasi*	
	Gradient Descent	*Conjugate Gradient*	*Gradient Descent*	*Conjugate Gradient*
0.001	*418.98*	***1.29***	*3.18*	***2.10***
0.005	*375.47*	***1.39***	*3.88*	***3.45***
0.01	*354.06*	***1.70***	*4.35*	***2.89***
0.1	*356.43*	***1.44***	***3.91***	*4.13*
0.2	*352.57*	***1.81***	*3.45*	***2.19***

Tabel 4 *Hasil evaluasi dengan menggunakan data testing untuk beberapa learning rate dengan 50 neuron tersembunyi.*

Learning Rate	*MSE (Means Square Error)*	
	Gradient Descent	*Conjugate Gradient*
0.001	*4627.75*	***27.49***
0.005	*2168.61*	***47.62***
0.01	*1607.21*	***9.95***
0.1	*2496.37*	***10.35***
0.2	*2022.84*	***31.37***

JST *Backpropagation* yang dioptimisasi oleh metode *Conjugate Gradient* menghasilkan MSE (*Mean Square Error*) terbaik yaitu sebesar 1.29 untuk data *training* dan sebesar 9.95 untuk data *testing*, sementara algoritma *Backpropagation* yang dioptimisasi dengan metode *Gradient Descent* menghasilkan MSE sebesar 105.45 untuk data *training* dan sebesar 915.89 untuk data *testing*. Gambar 3 (a) dan (b) menunjukkan masing-masing perbandingan dari data jumlah penderita COVID-19 yang sebenarnya dan hasil prediksi dengan menggunakan algoritma *Backpropagation* dioptimisasi dengan metode *Gradient Descent* dan algoritma *Backpropagation* yang dioptimisasi dengan metode *Conjugate Gradient* untuk data *training*. Gambar 4 (a) dan (b) menunjukkan masing-masing perbandingan dari data jumlah penderita COVID-19 yang sebenarnya dan hasil prediksi dengan menggunakan algoritma *Backpropagation* yang dioptimisasi dengan metode *Gradient Descent* dan algoritma *Backpropagation* yang dioptimisasi dengan metode *Conjugate Gradient* untuk data *testing*. Gambar 3 dan 4 memperlihatkan bahwa algoritma *Backpropagation* yang dioptimisasi dengan metode *Conjugate Gradient* memberikan hasil yang lebih baik dibandingkan dengan algoritma *Backpropagation* standar untuk memprediksi jumlah penderita COVID-19 di Kota Malang.

5 SIMPULAN

Dari hasil dan pembahasan maka dapat disimpulkan bahwa performa dari JST *Backpropagation* tergantung pada banyaknya *neuron* pada layer tersembunyi dan pembelajaran yang digunakan. *Neuron* tersembunyi yang terlalu banyak menyebabkan *over fitting* dan jika terlalu sedikit akan menghasilkan tingkat akurasi yang buruk. Peningkatan jumlah neuron pada *layer* tersembunyi menyebabkan waktu komputasi juga semakin lama. Peningkatan waktu durasi komputasi pada *Gradient Descent* tidak terlalu signifikan, begitu pula waktu komputasi untuk metode *Conjugate Gradient*. Laju pembelajaran yang terlalu kecil atau terlalu besar akan menghasilkan akurasi yang buruk. Oleh karena itu, perlu dipilih laju pembelajaran yang tepat sehingga dapat menghasilkan akurasi yang lebih baik. JST *Backpropagation* yang dioptimalkan oleh metode *Conjugate Gradient* memberikan hasil yang lebih baik dibandingkan dengan JST *Backpropagation* yang dioptimalkan oleh metode *Gradient Descent* untuk memprediksi jumlah peningkatan penderita COVID-19 di Kota Malang.

KEPUSTAKAAN

[1] Ibrahim, I. M., Abdelmalek, D. H., Elshahat, M. E., & Elfiky, *COVID-19 Spike-host Cell Receptor GRP78 Binding Site Prediction 80,* Journal of Infection,(5), 554–562. (2020).

[2] Pradanti, *Evaluation of Formal Risk Assessment Implementation of Middle East Respiratory Syndrome Coronavirus in 2018*, Jurnal Berkala Epidemologi, *7*(3), 197. (2019).

[3] Barda, N., Riesel, D., *Developing a COVID-19 Mortality Risk Prediction Model when Individual-level Data are not Available*, Nature Communications, *11*(1), 1–9. (2020)

DOI: 10.22236/teknoka.v5i.367
ISSN No. 2502-8782

[4] Kavadi, D. P., Patan, R., Ramachandran, M., & Gandomi, *Partial Derivative Nonlinear Global Pandemic Machine Learning Prediction of COVID 19, 139*. (2020).

[5] Wynants, L., Van Calster, B., *Prediction Models for Diagnosis and Prognosis of Covid-19: Systematic Review and Critical Appraisal*, Chaos, Solitons, and Fractals, *369*. (2020).

[6] Yi, Y., Lagniton, P. N. P., Ye, S., Li, E., & Xu, *COVID-19: What has been Learned and to be Learned about the Novel Coronavirus Disease*., International Journal of Biological Sciences, *16*(10), 1753–1766. (2020).

[7] Dona, Finky, *Komparasi Algoritma Conjugate gradient dan Gradient Descent pada MLPNN untuk Tingkat Pengetahuan Ibu*, Prosiding Konferensi Nasional Sistem & Informatika, 507–512. (2017).

[8] Bafitlhile, T. M., Li, Z., & Li, Q, *Comparison of Levenberg Marquardt and Conjugate Gradient Descent Optimization Methods for Simulation of Streamflow Using Artificial Neural Network*, Advances in Ecology and Environmental Research, *3*(2517–9454), 217–237. (2018).

[9] Thaheer, H, *Teknik Optimasi Lanjut*, Denpasar: Udayana University Press. (2019).

[10] Du, K. L., & Swamy, *Neural Networks in a Softcomputing Framework*, 1–566. (2006).

[11] Sudarsono, A. *Jaringan Syaraf Tiruan untuk Memprediksi Laju Pertumbuhan Penduduk Menggunakan Metode Backpropagation*, Media Infotama, 61–69 . (2016).

[12] Laurene, F., *Fundamentals of Neural Network, Architectures, Algorithm and Applications*. United State: Prentice Hall, inc. (1994).

DOI: 10.22236/teknoka.v5i.367

ISSN No. 2502-8782