Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 1, No. 11, November 2017, hlm. 1260-1269 http://j-ptiik.ub.ac.id

Pelatihan *Feedforward Neural Network* Menggunakan PSO untuk Prediksi Jumlah Pengangguran Terbuka di Indonesia

Bayu Septyo Adi[1], Dian Eka Ratnawati[2], Marji[3]

Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]bayusa69@gmail.com, [2]dian_ilkom@ub.ac.id, [3]marji@ub.ac.id

Abstrak

Pengangguran terbuka merupakan permasalahan yang dihadapi Indonesia setiap tahunnya. Jumlah pengangguran terbuka di Indonesia masih cukup tinggi. Banyak faktor yang mempengaruhi jumlah penganggura terbuka, salah satunya adalah tidak sebandingnya jumlah lapangan kerja dengan jumlah angkatan kerja. Semakin tinggi jumlah pengangguran, maka akan berdampak pula pada sektor lainnya, terutama sektor ekonomi karena jumlah pengangguran yang tinggi menyebabkan penurunan pendapatan nasional dan kemisikinan juga meningkat. Dengan memprediksikan jumlah pengangguran terbuka, data hasil prediksi diharapkan dapat membantu pemerintah dan instansi terkait untuk membuat program untuk mengurangi jumlah pengangguran terbuka di Indonesia. *Feedforward Neural Network* merupakan salah satu model dari jaringan saraf tiruan yang dapat diimplementasikan untuk melakukan prediksi. Algoritme *Particle Swarm Optimization* (PSO) dapat menggantikan algoritme *Backpropagation* untuk melatih *Feedforward Neural Network*. Hasil pengujian pada penelitian ini, nilai rata-rata *error* yang dihitung menggunakan *Average Forecast Error Rate* (AFER) sebesar 2.71399% . Dari nilai AFER yang dihasilkan dapat disimpulkan bahwa *Feedforward Neural Network* yang dilatih dengan PSO dapat digunakan untuk prediksi jumlah pengangguran terbuka di Indonesia dengan tingkat akurasi yang baik.

Kata kunci: *pengangguran terbuka, feedforward neural network (FFNN), particle swarm optimization (PSO), average forecasting error rate (AFER)*

Abstract

Open unemployment is a problem who faced by Indonesia in every year. In Indonesia, the number of an open unemployment is still in the high level. There are many factors influence the number of open unemployment, the one of that factor is the number of employement not comparable with the number of labor force. When the number of unemployment at the high level, it can influence the other sector, especially at the economy sector. Because of the number of unemployment is high, national income getting decrease and poorness getting increase. Prediction the number of open unemployment, can be expect to help government and other agence to decreasing the number of open unemployment in Indonesian. Feedforward Neural Network is model from artificial neural network which can be implemented for prediction. Backpropagation algorithm can be replaced by Particle Swarm Optimization Algorithm (PSO) for training Feedforward Neural Network . The result in this research, average value of error which is calculated by Average Forecast Error Rate (AFER) is 2.71399%. Based on value of AFER in this reaserch, Feedforward Neural Network trained by PSO method can be using for predicting the number of open unemployment in Indonesia with better accuracy.

Keywords: *open unemployment, feedforward neural network (FFNN), particle swarm optimization (PSO), average forecasting error rate (AFER)*

1. PENDAHULUAN

Indonesia merupakan salah satu negara berkembang di dunia dan juga salah satu negara yang memiliki penduduk terbesar didunia. Namun semakin banyaknya jumlah penduduk, semakin banyak pula permasalahan yang muncul di tengah-tengah masyarakat, terutama masalah pengangguran.

Pengangguran merupakan permasalahan di mana angkatan kerja yang tersedia tidak sebanding dengan lapangan kerja yang ada

(HM, 2016). Pada tahun 2015 lalu, survei yang dilakukan Badan Pusat Statistik (BPS) pada bulan Februari dan Agustus mencatat jumlah pengangguran terbuka mengalami kenaikan, dari 7.45 juta menjadi 7.56 juta jiwa (BPS, 2017).

Kenaikan jumlah pengangguran akan berdampak pada berbagai sektor. Sektor yang paling terasa adalah sektor ekonomi, karena secara langsung, jumlah pengangguran akan menurunkan pendapatan nasional dan juga meningkatkan angka kemiskinan (Harjanto, 2014).

Jumlah pengangguran terbuka dari tahun ke tahun tentu tidak dapat dipastikan sebelum adanya survei yang dilakukan oleh pihak terkait di mana prosesnya juga memakan waktu yang cukup lama. Permasalahan yang ada pada proses survei, dapat diatasi dengan menggunakan metode prediksi. Dengan memprediksikan jumlah penganggruan terbuka di Indonesia, data hasil prediksi dapat digunakan oleh pemerintah sebagai data acuan untuk membuat kebijakan mengatasi jumlah pengangguran terbuka di Indonesia.

Prediksi merupakan sebuah cara untuk memperkirakan kejadian dimasa depan berdasarkan data dimasa lalu dengan menggunakan model matematika (Heizer dan Barry, 2009). Salah satu model matematika yang dapat digunakan untuk prediksi data *time series* adalah model *feedforward neural network*. *Feedforward neural network* harus melalui proses pelatihan agar jaringan dapat menghasilkan data prediksi yang akurat. Dari beberapa penelitian, kebanyakan pelatihan *feedforward neural network* menggunakan algoritme *backpropagation*. *Backpropagation* merupakan algoritme yang berbasis *gradient*, sehingga permasalahan yang sering muncul di dalam penggunaan algoritme ini adalah sering terjebak di dalam lokal optimum (Bisi dan Goyal, 2015).

Untuk mengatasi kelemahan tersebut, pelatihan *feedforward neural network* dapat digantikan dengan algoritme *particle swarm optimization* (PSO) di mana algoritme ini termasuk di dalam algoritme kecerdasan berkelompok (*swarm intelligence*).

Penelitian yang menggunakan algoritme PSO untuk pelatihan *feedforward neural network* sudah banyak dilakukan. Rashid et al. (2015) menggunakan *feedforward neural network* yang dilatih dengan PSO untuk memprediksikan energi untuk menanamkan siklus daya gabungan. Saat proses pelatihan, nilai MSE yang diperoleh sebesar 1.019e-04 dan saat prediksi, nilai MSE yang diperoleh sebesar 0.0055. Penelitian lainnya untuk prediksi sistem *metering* gas yang dilakukan oleh Rosli et al. (2016) juga menggunakan PSO untuk pelatihan *feedforward neural network*. Penelitian ini menghasilkan tingkat *error* yang dihasilkan jaringan kurang dari 1%.

Berdasarkan dari permasalahan yang sudah disebutkan, fokus penelitian ini adalah bagaimana mengimplementasikan algoritme PSO untuk pelatihan *feedforward neural network* untuk memprediksikan jumlah pengangguran terbuka di Indonesia.

2. DASAR TEORI

2.1. Pengangguran Terbuka

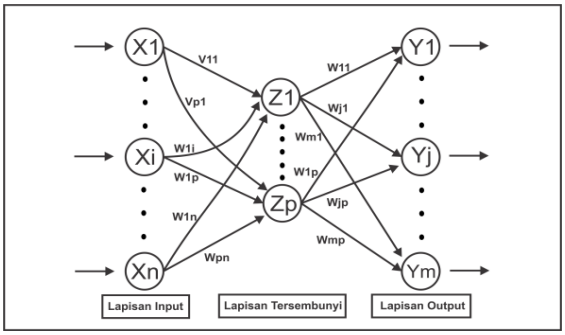
Pengangguran terbuka merupakan pengangguran yang muncul dikarenakan angkatan kerja yang tinggi namun kesempatan kerja yang ada rendah. Di dalam perekonomian, dampak yang diakibatkan adalah banyaknya tenaga kerja yang tidak mendapatkan pekerjaan, sehingga dalam jangka waktu yang lama, tenaga kerja ini akan menganggur secara waktu penuh (Harjanto, 2014).

2.2. Prediksi

Prediksi merupakan ilmu untuk memperkirakan apa yang terjadi dimasa mendatang dengan memproyeksikan data yang ada pada masa lalu. Teknik pendekatannya ada dua, yaitu secara kuantitatif dan kualitatif (Heizer dan Render, 2009).

2.3. *Feedforwad Neural Network* (FFNN)

Jaringan saraf tiruan adalah *neuron* yang saling terkoneksi berdasarkan model komputasional. Secara sederhana, jaringan saraf tiruan terdiri dari tiga lapisan utama, yaitu lapisan *input*, lapisan tersembunyi dan lapisan *output*. Jumlah *neuron* pada lapisan *input* selalu sama dengan jumlah fitur yang digunakan namun jumlah *neuron* pada lapisan *output* tergantung dari *output* yang diinginkan (Rashid et al., 2015).



Gambar 1. Arsitektur *Feedforward Neural Network*

Proses pelatihan maupun testing pada *feedforward neural network* adalah sebagai berikut (Siang, 2005):

1. Inisialisasikan bobot *Wjk* dan bias *Vij* secara acak.
2. Hitung keluaran pada *hidden* unit *Zj*, menggunakan Persamaan 1 dan 2.

$$Z\_net = V_{j0} + \sum_{i=1}^{n} X_i V_{ji} \quad (1)$$

$$Z\_j = f(Z\_net) = \frac{1}{1+e^{-z\_net_j}} \quad (2)$$

3. Hitung keluaran jaringan *Yk*, menggunakan Persamaan 3 dan 4.

$$Y\_net_k = W_{k0} + \sum_{j=1}^{p} Z_j W_{kj} \quad (3)$$

$$Y_k = f(y\_net_k) = \frac{1}{1+e^{-y\_net_k}} \quad (4)$$

2.4. *Particle Swarm Optimization* (PSO)

Particle Swarm Opitimization (PSO) adalah teknik optimasi yang menganalogikan perilaku sekawanan burung dan pembelajaran ikan yang dikembangkan oleh Eberhart dan Kennedy pada tahun 1990-an. Di dalam PSO, setiap individu akan menukar pengalaman hasil pembelajaran sebelumnya. PSO termasuk di dalam ranah penelitian kecerdasarn berkelompok (*swarm intelligence*). PSO memanfaatkan beberapa titik pencarian di dalam ruang solusi. Penggunaan PSO ditujukan untuk mengatasi permasalahan *non-linier* di mana variabel awalnya bersifat kontinyu. Di dalam pengembangannya, selain dapat mengatasi permasalahan kontinyu, PSO juga dapat mengatasi permasalahan diskret (Fukuyama, 2008).

Di dalam menyelesaiakan permasalahan menggunakan PSO, parameter-parameter yang digunakan adalah sebagai berikut (Jordehi dan Jasni, 2013):

1. *Swarm Size*

Ukuran *swarm* sangat berpengaruh pada performa PSO. Ukuran *swarm* yang terlalu kecil akan menyebabkan algoritme terjebak di dalam optimum lokal, namun jika terlalu besar juga akan menurunkan performansi algoritme. Tidak ada aturan khusus terkait ukuran *swarm*. Tetapi umunya, jika permasalahan yang dihadapi besar, maka ukuran *swarm* dapat diperbesar.

2. Teknik Inisialisasi

Hasil penelitian membuktikan bahwa inisialisasi partikel secara acak dapat memfasilitasi eksplorasi yang efektif dari berbagai ruang pencarian.

3. Nilai Kecepatan Maksimum

Nilai kecepatan maksimum akan mempengaruhi kinerja PSO. Jika kecepatan terlalu besar, partikel akan bergerak tak menentu dan akan mudah terjebak di dalam optimum *global*. Namun disisi lain, jika kecepatan partikel terlalu kecil, pergerakan partikel akan terlalu dibatasi yang mana dalam hal ini peningkatan waktu komputasi juga mungkin terjadi.

4. Koefisien Akselerasi

Nilai koefisien akselerasi merepresentasikan bobot akselerasi *stochastic*. Jika nilai ini terlalu besar, partikel akan bergerak terlalu cepat dan dapat terjebak di dalam optimal palsu. Sebaliknya jika nilainya terlalu kecil, partikel akan terlalu lambat bergerak dan dan waktu komputasi juga akan meningkat.

Bila nilai koefisien akselerasi kognitif (C1) meningkat, maka akan meningkatkan daya tarik partikel terhadap *PBest* dan akan mengurangi daya tarik terhadap *GBest*. Dari penelitian yang membahas tentang nilai koefisien akselerasi, diketahui bahwa nilai C1 = C2 = 2 umumnya dapat diterima untuk menyelesaikan permasalahan.

5. Kondisi Berhenti

Kriteria kondisi berhenti tergantung dari permasalahan yang dihadapi. Bisa jadi dengan iterasi yang kecil, algoritme sudah memberikan solusi, disisi lain tidak menutup kemungkinan jumlah iterasi yang diberikan lebih besar dikarenakan jumlah iterasi yang kecil belum memberikan solusi.

2.5. Penerapan Algoritme PSO

1. Kecepatan dan Posisi Partikel

Di dalam ruang pencarian *D*, *swarm* memiliki *N* partikel. Partikel ke *i-th* pada iterasi

ke *t-th* merupakan posisi partikel saat ini dan kecepatan saat ini. Posisi partikel disimbolkan dengan X dan kecepatan partikel disimbolkan dengan V (Li dan Liu, 2016).

$$X_i(t) = (X_{i1}(t), X_{i2}(t),...., X_{iD}(t)) \quad (5)$$

$$V_i(t) = (V_{i1}(t), V_{i2}(t),...., V_{iD}(t)) \quad (6)$$

2. *PBest* dan *GBest*

PBest (*Personal Best*) adalah posisi terbaik partikel yang disimpannya disebut juga dengan *local best position*. Sementara *GBest* (*Global Best*) adalah posisi terbaik keseluruhan partikel, disebut juga dengan *global position*. *PBest* dapat dinotasikan dengan Persamaan 7 sedangkan *GBest* dapat dinotasikan dengan Persamaan 8 (Li dan Liu, 2016).

$$P_i(t) = (P_{i1}(t), P_{i2}(t),....., P_{iD}(t)) \quad (7)$$

$$G(t) = (G_1(t), G_2(t),....., G_D(t)) \quad (8)$$

3. *Update* Kecepatan dan Posisi Partikel

Disetiap iterasi, kecepatan partikel dan posisi partikel akan terus di-*update*. Proses *update* kecepatan menggunakan Persamaan 9 sedangkan proses *update* posisi partikel menggunakan Persamaan 10 (Juneja dan Nagar, 2016).

$$V_{id}^{k+1} = \omega V_{id}^{k} + c_1 r_1 (pbest_{id}^{k} - x_{id}^{k}) + c_2 r_2 (gbest_{id}^{k} - x_{id}^{k}) \quad (9)$$

$$X_{id}^{k+1} = X_{id}^{k} + V_{id}^{k+1} \quad (10)$$

di mana:

V_{id}^{k} dan X_{id}^{k} = Kecepatan dan posisi partikel saat iterasi ke-*i*.

$pbest_{id}^{k}$ dan $gbest_{id}^{k}$ = *Pbest* dan *Gbest* iterasi ke-*i*.

ω = Bobot inersia iterasi ke-*i*.

c_1 dan c_2 = Merupakan nilai koefisien akselerasi.

r_1 dan r_2 = Nilai acak di dalam *range* [0, 1].

4. Nilai Bobot Inersia

Bobot inersia berfungsi untuk mengendalikan efek yang diberikan oleh kecepatan partikel. Jika nilai ω cenderung besar, maka algoritme akan meningkatkan kemampuan pencarian *global*. Sementara jika nilainya kecil, maka akan meningkatkan kemampuan pencarian parsial (Juneja dan Nagar, 2016).

Pada umumnya, algoritme PSO menggunakan penekatan *linier decreasing inertia weight* (LDIW) untuk mencari nilai ω. Namun, LDIW memiliki kelemahan, antara lain (Li dan Liu, 2016):

1. Di dalam tahap awal, kemampuan pencarian lokal tidak baik. Bahkan jika partikel sudah terlalu dekat dengan solusi optimal *global*, patikel selalu kehilangan solusi. Hal ini dikarenakan kecepatan partikel terlalu cepat.
2. Di dalam tahap akhir, kemapuan pencarian *global* menjadi buruk. Pada saat ini, algoritme dengan mudah terjebak di dalam optimum lokal.

Berdasarkan kelemahan dari LDIW, nilai bobot inersia pada penelitian ini dihitung menggunakan pendekatan *non-linier decreasing inertia weight* yang ditunjukkan pada Persamaan 11 (Li dan Liu, 2016).

$$\omega = \omega_{\max} - \frac{\sqrt{t+1} \times (\omega_{\max} - \omega_{\min})}{\sqrt{t_{\max} + 1}} \quad (11)$$

di mana:

ω = Nilai bobot inersia.

$\omega_{\max}$ = Nilai ω maksimum.

$\omega_{\min}$ = Nilai ω minimum.

$t_{\max}$ = Iterasi maksimum.

t = Iterasi saat ini.

2.6. Normalisasi Data

Normalisasi data diperlukan untuk untuk mempercepat pelatihan di dalam jaringan saraf tiruan (Haviluddin et al., 2016). Normalisasi data di dalam penelitian ini mengubah data ke dalam *range* [0.1, 0.9], yang mana prosesnya berdasarkan Persamaan 12 (Siang, 2005).

$$X' = \frac{0.8(X - a)}{b - a} + 0.1 \quad (12)$$

di mana:

X' = Data normalisasi.

X = Data asli.

a = Data minimum.

b = Data maksimum.

2.7. Denormalisasi Data

Proses denormalisasi data adalah proses untuk mengubah data yang ada di dalam *range* [0.1, 0.9] ke dalam nilai riil. Proses denormalisasi data menggunakan Persamaan 13 (Siang, 2005).

$$X = \frac{(X'-0.1)\times(x.\max - x.\min)}{0.8} + x.\min \quad (13)$$

di mana:
X = Data denormalisasi.
X' = Data normalisasi.
x.max = Nilai maksimum data aktual.
x.min = Nilai minimum data aktual.

2.8. Perhitungan Kesalahan Prediksi

Besarnya *error* yang dihasilkan oleh suatu metode merupakan kriteria penolakan terhadap metode prediksi yang digunakan. Di dalam perhitungan *error* ini pula, dapat dilihat *goodness of fit*, yaitu kemampuan sebuah metode seberapa jauh dapat melakukan prediksi pada data yang sudah diketahui (Syukriyawati, 2015).

Di dalam penelitian ini, perhitungan kesalahan prediksi menggunakan AFER (*Average Forecasting Error Rate*). Perhitungan AFER menggunakan Persamaan 14 (Syukriyawati, 2015).

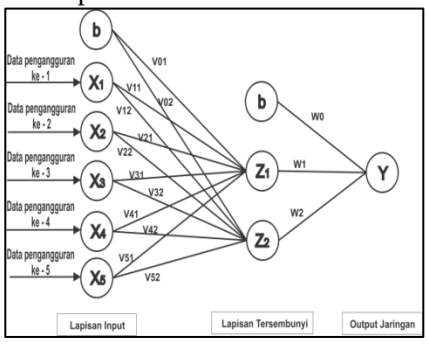
$$AFER = \frac{\sum\left|(Ai - Fi)/Ai\right|}{n} \times 100\% \quad (14)$$

di mana:
AFER = Nilai *Average Forecasting Error Rate*.
Ai = Data aktual ke-*i*.
Fi = Data hasil prediksi ke-*i*.
n = Jumlah data.

3. METODOLOGI PENELITIAN

3.1. Arsitektur Jaringan Saraf Tiruan

Di dalam penelitian ini, arsitektur jaringan saraf tiruan yang digunakan terdiri dari 1 buah lapisan *input*, 1 buah lapisan tersembunyi dan satu 1 buah lapisan *output*. Secara umum, arsitektur jaringan saraf tiruan yang digunakan ditunjukkan pada Gambar 2.

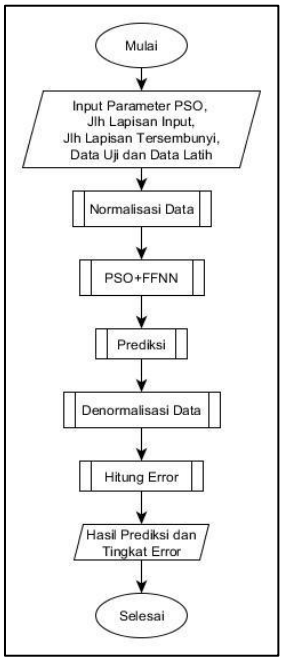


Gambar 2. Arsitektur Jaringan Saraf Tiruan Penelitian

Dari Gambar 2 dijelaskan bahwa di dalam arsitektur jaringan saraf tiruan pada penelitian ini, lapisan *input* memiliki jumlah *neuron* yang disesusaikan dengan pola data yang digunakan. Satu buah *neuron* pada lapisan *input* mewakili data pengangguran 1 tahun. Bias pada lapisan *input* dan lapisan tersembunyi yang digunakan sama-sama 1.

3.2. Alur Penyelesaian Masalah Menggunakan PSO-*Feedforward Neural Network*

Secara umum, proses penyelesaian permasalahan pelatihan *feedforward neural network* untuk prediksi jumlah pengangguran terbuka ditunjukkan pada Gambar 3.



Gambar 3. Diagram Alir Pelatihan *Feedforward Neural Network* Menggunakan PSO

Sebelum proses pelatihan dilakukan, ada dua hal yang harus terdefinisi terlebih dahulu. Dua hal tersebut adalah sebagai berikut:

1. Representasi partikel

Representasi partikel berupa *real code particle swarm optimization*. Nilai-nilai yang ada di dalam partikel digunakan sebagai bobot dan bias jaringan . Besarnya dimensi partikel dihitung menggunakan Persamaan 15 (Asriningtias, Dachlan dan Yudaningtias, 2015).

$$w = (m+1)n + (n+1)q \quad (15)$$

di mana:
w = Bobot dan bias jaringan.
m = Jumlah unit *input*.

n = Jumlah unit *hidden*.
q = Jumlah unit *output*.

Berdasarkan arsitektur jaringan yang ditunjukkan pada Gambar 2, representasi partikel yang didapatkan adalah sebagai berikut:

$$w = (m+1)n + (n+1)q = (5+1)\times 2 + (2+1)\times 1 = 15$$

X1	X2	X3	X4	X5	X6	X7	X8	X9	X10
V01	V02	V11	V12	V21	V22	V31	V32	V41	V42
X11	X12	X13	X14	X15					
V51	V52	W0	W1	W2					

Keterangan:

Bias dari lapisan *input* ke lapisan tersembunyi.

Bobot dari lapisan *input* ke lapisan tersembunyi.

Bias dari lapisan tersembunyi ke lapisan *output*.

Bobot dari lapisan tersembunyi ke lapisan *output*.

2. Fungsi fitness

Selama proses pelatihan, setiap partikel membawa satu solusi. Untuk melihat seberapa baik solusi yang dibawa setiap partikel, maka harus dilakukan evaluasi partikel melalui fungsi *fitness* (Asriningtias, Dachlan dan Yudaningtias, 2015).

Fungsi *fitness* yang digunakan selama pelatihan menggunakan ditunjukkan pada Persamaan 16 (Rashid et al., 2015).

$$Fitness = \frac{1}{MSE} = \frac{1}{\frac{1}{n}\sum_{i=1}^{N}\left(\lambda(\tau_i;x) - y_i\right)^2} \qquad (16)$$

di mana:
Fitness = Nilai *fitness*.
$\lambda(\tau_i;x)$ = Target aktual.
y_i = Data prediksi.
n = Jumlah data latih.

Setelah representasi partikel dan fungsi fitness sudah jelas terdefinisi, proses pelatihan *feedforward neural network* menggunakan PSO sudah dapat dilakukan. Langkah-langkah pelatihan *feedforward neural network* menggunakan PSO (PSO+FFNN) adalah sebagai berikut (Rashid et al., 2015):

1. Inisialisasi semua parameter PSO dan juga jumlah lapisan *input* dan lapisan tersembunyi pada jaringan saraf tiruan.
2. Inisialisasi posisi partikel sebagai bobot dan bias jaringan. Selain itu, inisialisasikan juga kecepatan partikel.
3. Hitung nilai *fitness* setiap partikel menggunakan Persamaan 16. Nilai *fitness* didapatkan dari proses *feedforward neural network*.
4. *Update PBest* dan *GBest* berdasarkan nilai *fitness* yang dihasilkan setiap partikel.
5. Jika kondisi berhenti belum tercapai, ulangi langkah 3 dan 4.
6. Jika kondisi berhenti terpenuhi, gunakan *GBest* hasil pelatihan pada iterasi terakhir sebagai bobot dan bias jaringan saraf tiruan.
7. Lakukan proses prediksi menggunakan *feedforward neural network*.

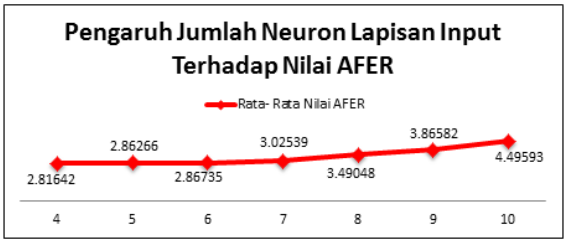
3.3. Data Penelitian

Pada penelitian ini, data yang digunakan adalah data jumlah pengangguran terbuka di Indonesia dari tahun 1986-2016 yang diambil dari *website* Badan Pusat Statistik (BPS).

4. HASIL PENGUJIAN DAN ANALISIS

4.1. Hasil Pengujian Jumlah Neuron Pada Lapisan Input.

Pengujian ini bertujuan untuk mengetahui jumlah *neuron* pada lapisan *input* yang menghasilkan nilai AFER terkecil. Jumlah *neuron* yang diuji dimulai dari 4 sampai 10 sedangkan jumlah *neuron* pada lapisan tersembunyi adalah 2. Jumlah partikel yang digunakan 20, iterasi 150, nilai C1=C2= 2, ω_{max} 0.9, nilai ω_{min} 0.4, data latih 32 dan data uji 9. Pengujian dilakukan sebanyak 10 kali.



Gambar 4. Grafik Pengujian Jumlah *Neuron* pada Lapisan *Input*

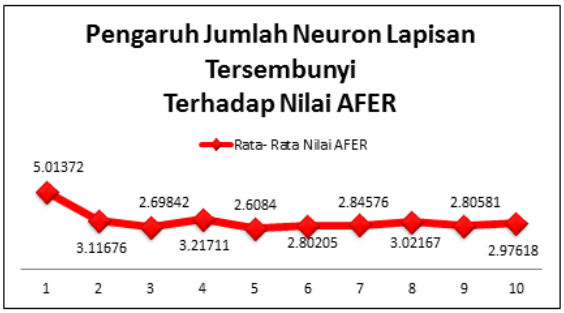
Gambar 4 menunjukkan bahwa jumlah *neuron* pada lapisan *input* dari 4 sampai 10 nilai AFER yang dihasilkan jaringan cenderung

mengalami kenaikan. Kenaikan nilai AFER secara signifikan terjadi saat jumlah *neuron* pada lapisan *input* sebesar 7 sampai 10 dengan kenaikan nilai AFER disetiap jumlah *neuron* sebesar 0.4. Hal ini dikarenakan semakin besar jumlah *neuron* pada lapisan *input*, maka jaringan kurang dapat memproses informasi yang diberikan dengan baik. Selain itu, jaringan juga kurang dapat merepresentasikan pola prediksi sehingga nilai AFER yang dihasilkan mangalami kenaikan cukup besar sehingga hasil prediksi kurang mendekati nilai aktual.

Sementara itu, ketika jaringan menggunakan jumlah *neuron* pada lapisan *input* sebesar 4 sampai 6, nilai AFER yang dihasilkan jaringan mengalami kenaikan namun tidak terlalu siginifikan. Dengan kata lain, jumlah neuron pada lapisan *input* di dalam rentang 4 sampai 6 di dalam penelitian ini dapat memproses informasi dengan lebih baik dan juga dapat merepresentasikan pola prediksi sehingga hasil prediksi dapat mendekati nilai aktual.

4.2. Hasil Pengujian Jumlah Neuron Pada Lapisan Tersembunyi.

Pengujian ini bertujuan untuk mengetahui jumlah *neuron* pada lapisan tersembunyi yang menghasilkan nilai AFER terkecil. Jumlah *neuron* yang diuji dimulai dari 1 sampai 10 sedangkan jumlah *neuron* pada lapisan *input* yang digunkan adalah 4. Jumlah partikel yang digunakan 20, iterasi 150, nilai C1=C2= 2, ω_{max} 0.9, nilai ω_{min} 0.4, data latih 32 dan data uji 9. Pengujian dilakukan sebanyak 10 kali.



Gambar 5. Grafik Pengujian Jumlah *Neuron* pada Lapisan Tersembunyi

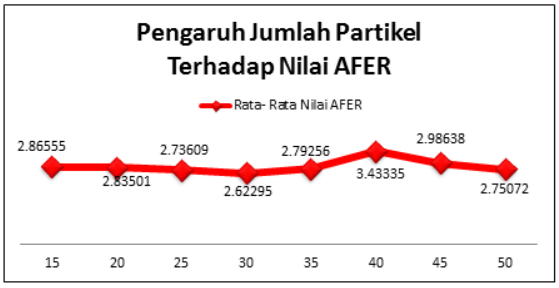
Gambar 5 menunjukkan bahwa ketika jumlah *neuron* kurang dari 5, rata-rata nilai AFER yang dihasilkan jaringan cukup tinggi. Kondisi ini disebut dengan *underfitting*. *Underfitting* adalah kondisi di mana *neuron* pada lapisan tersembunyi terlalu sedikit sehingga jaringan kesulitan untuk mendeteksi seluruh sinyal pada sekumpulan data yang rumit (Karsoliya, 2012).

Demikian juga ketika jumlah *neuron* lebih besar dari 5, rata-rata nilai AFER yang dihasilkan tidak lebih kecil saat jumlah *neuron* sama dengan 5, namun nilai rata-rata AFER cenderung kecil. Kondisi ini disebut dengan *overfitting*. Hal ini dikarenakan jaringan dapat mendeteksi seluruh sinyal pada sekumpulan data yang rumit, namun hanya pada saat proses pelatihan saja. Sementara ketika proses prediksi, pola data prediksi sulit dideteksi.

Agar jaringan tidak mengalami kondisi *underfitting* maupun *overfitting*, maka perlu dilakukan uji coba untuk mencari jumlah *neuron* pada lapisan tersembunyi yang menghasilkan nilai AFER terkecil. Hal ini dikarenakan sampai saat ini tidak ada persamaan untuk menghitung berapa banyak jumlah *neuron* pada lapisan tersembunyi yang baik untuk jaringan (Karsoliya, 2012).

4.3. Hasil Pengujian Jumlah Partikel

Pengujian ini bertujuan untuk mengetahui jumlah partikel yang menghasilkan nilai AFER terkecil. Jumlah partikel yang diujikan dimulai dari 15 sampai 50 dengan kelipatan 5. Jumlah *neuron* pada lapisan *input* yang digunakan 4, jumlah *neuron* pada lapisan tersembunyi yang digunakan adalah 5. Jumlah iterasi 150, nilai C1=C2= 2, ω_{max} 0.9, nilai ω_{min} 0.4, data latih 32 dan data uji 9. Pengujian dilakukan sebanyak 10 kali.



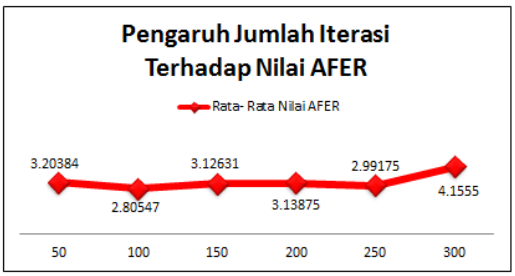
Gambar 6. Grafik Pengujian Jumlah Partikel

Gambar 6 menunjukkan bahwa saat jumlah partikel sebesar 15 sampai 30, nilai AFER yang dihasilkan cenderung menurun. Hal ini menunjukkan bahwa pencarian solusi sudah didapat berdasarkan jumlah partikel di dalam rentang tersebut. Saat proses pelatihan, semakin besar jumlah partikel yang diberikan, maka calon solusi yang diberikan juga akan semakin banyak (Eliantara, Cholissodin dan Indriati, 2016). Namun, saat jumlah partikel 35 sampai

40, nilai rata-rata AFER naik secara signifikan. Kemudian kembali turun saat jumlah partikel yang diberikan sebesar 45 sampai 50. Hal ini menunjukkan bahwa pada jumlah partikel tertentu, nilai rata-rata AFER yang dihasilkan jaringan akan tinggi, meskipun jumlah partikel yang digunakan cukup besar.

4.4. Hasil Pengujian Iterasi

Pengujian ini bertujuan untuk mengetahui jumlah iterasi yang menghasilkan nilai AFER terkecil. Jumlah iterasi yang diujikan dimulai dari 50 sampai 300 dengan kelipatan 50. Jumlah *neuron* pada lapisan *input* yang digunakan 4, jumlah *neuron* pada lapisan tersembunyi yang digunkan adalah 5. Jumlah partikel 30, nilai C1=C2= 2, ω_{max} 0.9, nilai ω_{min} 0.4, data latih 32 dan data uji 9. Pengujian dilakukan sebanyak 10 kali.



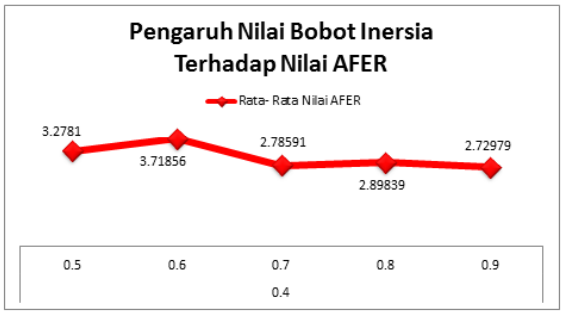
Gambar 7. Grafik Pengujian Jumlah Iterasi

Gambar 7 menunjukkan bahwa pengaruh jumlah iterasi terhadap nilai AFER tidak memiliki pola dan juga jumlah iterasi yang diberikan semakin besar, belum tentu menghasilkan nilai rata-rata AFER yang kecil. Hal ini dapat dilihat saat iterasi sebesar 50, 150, 200 dan 300, nilai rata-rata AFER yang dihasilkan cenderung besar, sementara saat iterasi sebesar 100 dan 250, nilai rata-rata AFER cenderung kecil.

Pada proses pelatihan dengan menggunakan PSO, proses yang ada di dalamnya melibatkan proses *global optimization*. Proses *global optimization* adalah proses yang mana sifatnya acak. Sehingga, setiap kali proses pelatihan, nilai *error* yang dihasilkan akan selalu berbeda meskipun jumlah iterasinya sama (Asriningtias, Dachlan dan Yudaningtyas, 2015).

4.5. Hasil Pengujian Nilai Bobot Inersia

Pengujian ini bertujuan untuk mengetahui nilai bobot inersia yang menghasilkan nilai AFER terkecil. Nilai bobot inersia yang diujikan berada di dalam *range* [0.4 , 0.9]. Jumlah *neuron* pada lapisan *input* yang digunakan 4, jumlah *neuron* pada lapisan tersembunyi yang digunkan adalah 5. Jumlah partikel 30, jumlah iterasi 100, nilai C1=C2= 2, data latih 32 dan data uji 9. Pengujian dilakukan sebanyak 10 kali.

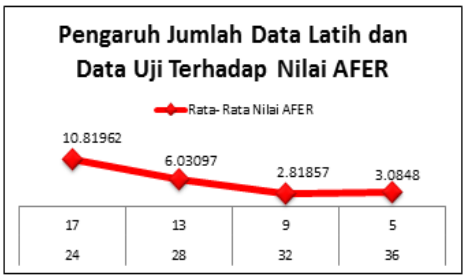


Gambar 8. Grafik Pengujian Jumlah Iterasi

Gambar 8 menunjukkan bahwasannya nilai AFER saat nilai bobot inersia hasil kombinasi dari ω_{max} 0.5 dan ω_{min} 0.4 dan ω_{max} 0.6 dan ω_{min} 0.4 cenderung besar. Hal ini dikarenakan rentang ω_{max} dan ω_{min} saat kombinasinya 0.4 dan 0.5 atau 0.4 dan 0.6 cenderung kecil, maka nilai bobot inersia juga akan kecil sehingga menyebabkan kecepatan partikel menurun. Dikarenakan kecepatan partikel menurun, daerah eksplorasi *swarm* menjadi kecil dan solusi optimal akan lebih cepat ditemukan. Sementara itu, nilai rata-rata AFER saat nilai hasil kombinasi dari ω_{max} 0.7, 0.8, 0.9 dan ω_{min} 0.4 cenderung kecil. Hal ini dikarenakan nilai bobot inersia hasil kombinasi dari nilai-nilai tersebut cenderung besar. Jika nilai bobot inersia besar, maka kecepatan partikel juga akan meningkat sehingga daerah eksplorasi *swarm* menjadi lebih besar.

4.6. Hasil Pengujian Jumlah Data Latih dan Data Uji

Pengujian ini bertujuan untuk mengetahui nilai jumlah data latih dan data uji yang menghasilkan nilai AFER terkecil. Jumlah data latih dan data uji yang diujikan berada di dalam *range* [5, 36]. Jumlah *neuron* pada lapisan *input* yang digunakan 4, jumlah *neuron* pada lapisan tersembunyi yang digunkan adalah 5. Jumlah partikel 30, jumlah iterasi 100, nilai C1=C2= 2, nilai ω_{max} 0.9 dan ω_{min} 0.4 Pengujian dilakukan sebanyak 10 kali.



Gambar 9. Grafik Pengujian Jumlah Iterasi

Gambar 9 menunjukkan bahwasannya nilai rata-rata AFER cenderung menurun. Hal ini dapat dilihat saat jumlah data latih yang digunakan sebanyak 24, 28 dan 32. Nilai AFER yang tadinya 10.81962% saat data latih yang digunakan sebanyak 24 data, turun secara signifikan menjadi 2.81857% saat data latih yang digunakan sebanyak 32. Namun, saat data latih yang digunakan sebanyak 36 data, nilai rata-rata AFER naik namun tidak signifikan. Berdasarkan hal tersebut, dapat diketahui bahwa jumlah data latih dan data uji yang proporsional dapat menghasilkan nilai AFER yang kecil.

4.7. Analisis Hasil

Berdasarkan hasil pengujian, Tabel 1 menunjukkan nilai yang optimal untuk setiap parameter yang digunakan di dalam pelatihan *feedforward neural network* menggunakan PSO untuk prediksi jumlah pengangguran terbuka di Indonesi. Tabel 2 menunjukkan hasil pengujian parameter optimal.

Tabel 1. Parameter Optimal Hasil Pengujian

No	Parameter	Nilai
1	Jumlah *neuron* lapisan *input*	4
2	Jumlah *neuron* lapisan tersembunyi	5
3	Jumlah partikel	30
4	Jumlah Iterasi	100
5	ω max	0.9
	ω min	0.4
6	Jumlah data latih	32
7	Jumlah data uji	9

Tabel 2. Hasil Pengujian Parameter Optimal

Uji Coba Ke -	AFER (%)	Waktu Eksekusi
1	2.3722	1.084
2	3.1976	0.966
3	2.3273	0.975
4	2.6872	0.927
5	3.4435	0.934
6	2.1542	0.992
7	2.4555	0.947
8	2.5611	0.938
9	2.2718	1.08
10	3.6695	0.978
Rata-Rata	**2.71399**	**0.9821**

Berdasarkan Tabel 1, nilai AFER terkecil didapat pada uji coba ke-6 dengan nilai AFER sebesar 2.1542% dan waktu komputasi sebesar 0.992 detik. Sementara nilai AFER terbesar didapat pada uji coba ke-10 dengan nilai AFER sebesar 3.6695% dan waktu komputasi sebesar 0.978 detik.

Secara umum, pelatihan *feedforward neural network* menggunakan PSO memberikan hasil prediksi yang cukup baik dan juga tidak memakan waktu yang cukup lama saat proses pelatihan. Dengan menggunakan kombinasi nilai disetiap parameter yang tepat, maka hasil prediksi juga akan semakin baik. Hal ini ditunjukkan dengan nilai AFER yang dihasilkan jaringan dari 10 kali pengujian, nilai AFER yang dihasilkan berkisar antara 2.1% sampai 3.7%.

4.8. Hasil Prediksi Berdasarkan Data Uji

Setelah didapat seluruh parameter optimal dan mengetahui rata-rata kesalahan prediksi, terakhir adalah melihat hasil prediksi pada data uji. Hasil prediksi berdasarkan data uji ditunjukkan pada Tabel 3.

Tabel 3. Hasil Prediksi Berdasarkan Data Uji

Tahun	Data Aktual	Hasil Prediksi	Selisih
2012 Februari	7757831	7452779	305052
2012 Agustus	7344866	7432389	87523
2013 Februari	7240897	7316236	75339
2013 Agustus	7410931	7198080	212851
2014 Februari	7147069	7153671	6602
2014 Agustus	7244905	7144622	100283
2015 Februari	7454767	7127408	327359
2015 Agustus	7560822	7137351	423471
2016 Februari	7024172	7173897	149725

5. PENUTUP

5.1. Kesimpulan

1. Algoritme PSO dapat diimplementasikan untuk pelatihan *feedforward neural network*. Proses pelatihan terjadi saat perhitungan nilai *fitness* setiap partikel, dikarenakan perhitungan *fitness* menggunakan langkah-langkah yang ada pada *feedforward neural network*.
2. Dari pengujian setiap parameter yang diperlukan untuk pelatihan *feedforward neural network* menggunakan PSO, didapatkan nilai-nilai parameter yang optimal, antara lain jumlah *neuron* pada lapisan *input* sebesar 4, jumlah *neuron* pada lapisan tersembunyi sebesar 5, jumlah partikel sebesar 30, jumlah iterasi sebesar 100, nilai ω max 0.9, nilai ω min 0.4 dan nilai C1 = C2 = 2, jumlah data latih 32 dan jumlah data uji 9.
3. Hasil pengujian berdasarkan parameter optimal didapatkan nilai AFER terkecil sebesar 2.1542% dan nilai AFER terbesar sebesar 3.6695%. Secara rata-rata, nilai AFER yang dihasilkan algoritme sebesar 2.71399%.

6. DAFTAR PUSTAKA

Asriningtias, S. R., Dachlan, H.S. & Yudaningtias E., 2015. Optimasi Training Neural Neural Network Menggunakan Hybrid Adaptive Mutation PSO-BP. *Jurnal EECCIS*. 9 (1) 79-84.

Badan Pusat Statistik., 2017. [data] Jumlah Pengangguran Terbuka di Indonesia (online). Tersedia di:https://www.bps.go.id/linkTabelStatis/view/id/972.

Bisi M. & Goyal N. K., 2015. Predicting Cummulative Number of Failures in Software using an ANN-PSO based approach. *IEEE*, pp. 9 – 14.

Eliantara F., Cholissodin I., Indriati., 2016. Optimasi Pemenuhan Kebutuhan Gizi Keluarga Menggunakan Particle Swarm Optimization. *Prosiding Seminar Nasional Riset Terapan (SNRT)*, 9-10 Nopember, Politeknik Negeri Banjarmasin.

Fukuyama, Y., 2008. *Modern Heuristic Optimization Techniques*. Institute of Electrical an Electronics Engineers Inc.

Harjanto, T., 2014.Pengangguran dan Pembangunan Nasional. .*Jurnal Ekonomi* , 2 (2).

Heizer, J. & Reizer B., 2009. *Manajemen Operasi*. Jakarta:Salemba Empat.

HM, Muhdar., 2015. Potret Ketengakerjaan, Pengangguran dan Kemiskinan di Indonesia:Masalah dan Solusi. *Al-Buhuts* , 11 (1) 42 -66.

Jordehi, A. R. & Jasni J., 2013. Parameter Selection in Particle Swarm Optimization:A survei. *Journal of Experimental & Theoritical Artificial Intelligence*. 25 (4) 527 -542.

Juneja M. & Nagar S.K., 2016. Particle Swarm Optimization Algorithm and Its Parameters:A review. *International Conference on Control, Computing, Communication and Materials (ICCCCM)* .

Karsoliya S., 2012. Approximating Number of Hidden Layer Neurons in Multiple Hidden Layer BPNN Architecture. *International Journal of Enginnering Trends and Technology*. 3 (6) 714 -717.

Li C. & Liu X., 2016. An Improved PSO-BP Neural Network and Its Application to Earthquake Prediction. *28th Chinese Control and Decision Conference* pp. 3434 -3438.

Rashid et al., 2015. Energy Prediction of a Combined Cycle Power Plant Using a Particle Swarm Optimization Trained FeedForward Neural Network. *International Conference on Mechanical Engineering, Automation and Control Systems (MEACS)*.

Rosli N. S., Ibrahim R., & Ismail I., 2016. Neural Network Model with Particle Swarm Optimization for Prediction in Gas Metering Systems. *IEEE*.

Syukriyawati G., 2015. Implementasi Metode Average-Based Fuzzy Time Series Models Pada Prediksi Jumlah Penduduk Provinsi DKI Jakarta. S1. Universitas Brawijaya.

Siang, J. J., 2005. *Jaringan Saraf Tiruan dan Pemrogramannya Menggunakan MATLAB*. Yogyakarta:Andi.