Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 2, No. 9, September 2018, hlm. 3333-3344 http://j-ptiik.ub.ac.id

Optimasi *K-Nearest Neighbour* Menggunakan *Particle Swarm Optimization* pada Sistem Pakar untuk *Monitoring* Pengendalian Hama pada Tanaman Jeruk

Kukuh Wiliam Mahardika[1], Yuita Arum Sari[2], Achmad Arwan[3]

Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]kukuhwiliam17@gmail.com, [2]yuita@ub.ac.id, [3]arwan@ub.ac.id

Abstrak

Jeruk di Indonesia merupakan salah satu komoditas nasional memiliki potensi daya saing yang tinggi dalam perekonomian lokal hingga ke luar negeri. Namun produksi Jeruk Indonesia sejak 2006 sampai 2015 mengalami penurunan. Salah satu penyebab penurunan ini adalah hama. Oleh karena itu dibutuhkan suatu sistem yang dapat memonitor pengendalian hama pada tanaman Jeruk. Metode PSO-KNN merupakan salah satu metode yang dapat digunakan untuk menyelesaikan masalah klasifikasi dengan banyak fitur. Metode ini adalah gabungan dari 2 metode yaitu *K-Nearest Neighbour* dan *Particle Swarm Optimization*. *K-Nearest Neighbour* (KNN) digunakan untuk mengklasifikasikan hama berdasarkan perhitungan kemiripan dari data yang sudah ada. *Particle Swarm Optimization* (PSO) digunakan untuk melakukan optimasi nilai *k* dan seleksi fitur pada *dataset* KNN dan kemudian mengevaluasi akurasi yang dihasilkan pada KNN. Dari hasil pengujian yang telah dilakukan dapat diambil kesimpulan bahwa nilai dari parameter PSO yang terbaik yaitu iterasi sebesar 151, *popsize* yaitu 25, nilai *c1* yaitu 1, nilai *c2* yaitu 1,2 dan nilai *w* yaitu 0,9. Terjadi peningkatan akurasi pada sebelum dan sesudah optimasi yaitu akurasi tertinggi pada KNN mencapai 90%, dan akurasi tertinggi pada PSO-KNN mencapai 96,25%. Adanya peningkatan akurasi menunjukkan bahwa PSO mampu memperbaiki kekurangan yang ada pada KNN.

Kata kunci: jeruk, hama, seleksi fitur, *KNN, PSO-KNN, Particle Swarm Optimization*

Abstract

Orange in Indonesia is one of the national commodities have the potential of high competitiveness in the local economy to abroad. The production of Indonesian Orange from 2006 to 2015 has decreased. One of the causes of this decline is pests. Therefore we need a system that can identify pests in citrus plants. The PSO-KNN method is one method that can be used to solve classification problems with many features. This method is a combination of 2 methods namely K-Nearest Neighbour and Particle Swarm optimization. K-Nearest Neighbors (KNN) are used to classify pests based on similarity calculations from existing data. Particle swarm optimization (PSO) is used to perform k value optimization and feature selection on KNN dataset and then evaluate the accuracy generated on KNN. From the results of tests that have been done can be concluded that the value of the best PSO parameter iteration is 151, popsize is 25, the value of c1 is 1, the value of c2 is 1.2 and w is 0.9. There was an increase in accuracy before and after optimization that is the highest accuracy of KNN reaches 90%, and the highest accuracy of PSO-KNN reached 96.25%. Improved accuracy indicates that the PSO algorithm is able to correct the deficiency that exist in KNN.

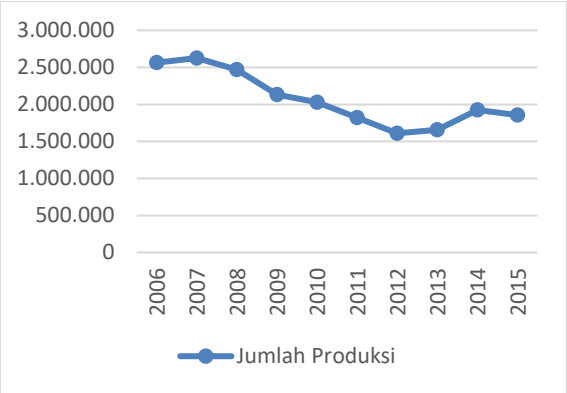
Keywords: *orange, pest, feature selection, KNN, PSO-KNN, Particle Swarm Optimization*

1. PENDAHULUAN

Jeruk merupakan jenis buah subtropika yang terdiri dari beberapa jenis yaitu Jeruk Manis, Keprok, Siam, Nipis, Pamelo, dan Purut (Endarto, 2016). Jeruk di Indonesia merupakan salah satu komoditas nasional memiliki potensi daya saing yang tinggi dalam perekonomian lokal hingga ke luar negeri. Upaya meningkatkan daya saing Jeruk lokal akan menunjang ekonomi masyarakat secara nasional melalui beberapa aspek yang salah satunya

adalah perbaikan teknologi budidaya untuk perbaikan produktivitas (Ichsan, 2016).

Produksi Jeruk Indonesia cenderung mengalami penurunan. Penurunan tersebut sesuai dengan Gambar 1 dari BPS (Badan Pusat Statistik) yang menunjukkan penurunan produksi Jeruk dari tahun 2006 sampai 2015.



Gambar 1. Data Produksi Jeruk Tiap Tahun

Salah satu penyebab penurunan ini adalah hama. Hal ini dibuktikan dengan beberapa kasus yang terjadi sebelumnya yaitu kasus sentra jeruk di Subang terserang hama gingga mengakibatkan kerugian hasil panen hingga puluhan juta rupiah (Naga, 2011). Bahkan pada kasus lainnya yang terjadi di medan, terjadi serangan hama yang berlangsung cukup lama dan kerugiannya sangat besar hingga menyebabkan para petani beralih budidaya kopi (Priadi, 2015).

Untuk mengatasi permasalahan hama tersebut maka dibuatlah sistem pakar hama tanaman jeruk yang bertujuan untuk membantu pekerjaan para petani dalam melakukan pengamatan langsung di lapangan. Sehingga masyarakat dapat lebih mudah untuk mengetahui kondisi tanaman jeruk secara akurat dan dapat melakukan penanganan yang benar sehingga menghasilkan jeruk dengan kualitas yang bagus.

Salah satu metode yang dapat digunakan untuk membantu memonitor pengendalian hama jeruk adalah metode *K-Nearest Neighbor* (K-NN). Algoritme K-NN adalah suatu metode *supervised* yang bertujuan mendapatkan pola baru suatu data dengan menghubungkan pola data sebelumnya dengan yang baru untuk mengklasifikasikan data ke dalam beberapa macam kelas berdasarkan atribut yang ada (Krisandi, 2013). Hal tersebut dapat digunakan untuk mengidentifikasi pengendalian hama pada tanaman jeruk. Metode K-NN ini dipilih dikarenakan memiliki kelebihan yaitu efektif diterapkan pada data dengan jumlah besar dan tangguh terhadap suatu data latih yang *noise* yang merupakan sebuah data yang memiliki *range* nilai paling jauh dibanding data lainnya tetapi dapat mengganggu struktur data yang ada. Selain kelebihan yang dimiliki, pada K-NN juga terdapat kelemahan yaitu kurang optimal dalam menentukan nilai *k* yang merupakan jumlah tetangga terdekat dan harus menentukan atribut yang akan dipilih atau seleksi fitur guna mendapat hasil terbaik. (Mutrofin, 2014).

Untuk mengatasi kekurangan tersebut diperlukan sebuah solusi perbaikan untuk mengoptimalkan klasifikasi K-NN. Salah satu solusi yang bisa diterapkan adalah dengan melakukan optimasi dengan menggunakan metode *Particle Swarm Optimization* (PSO). Dalam PSO setiap solusi ada dalam ruang pencarian dipandang sebagai partikel yang mana setiap partikel memiliki nilai *fitness* untuk dioptimalkan, dan memiliki kecepatan untuk perpindahan partikel (Hanafi, 2016). PSO memiliki beberapa kelebihan yaitu sedikit parameter, mudah diterapkan, konvergensi yang cepat, dan sederhana sehingga PSO banyak diterapkan pada optimasi fungsi, optimasi metode konvensional dan klasifikasi pola (Hasanuddin, 2016).

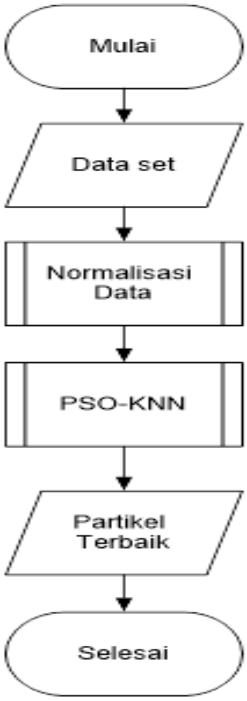
Salah satu penelitian mengenai penerapan seleksi fitur menggunakan PSO-KNN adalah penelitian pada masalah diagnosis pengelompokan mikrokalsifikasi dalam mamografi (Zyout, 2011). Dalam penelitian tersebut menggunakan dataset yang terdiri dari 34 fitur. Dari penelitian yang telah dilakukan, hasil akurasi KNN yang dioptimasi lebih tinggi dari pada tanpa optimasi dengan peningkatan akurasi tertinggi mencapai 38%, yaitu dari 56% menjadi 94%.

Berdasarkan latar belakang yang sudah dipaparkan, maka dipilih metode K-NN dengan optimasi PSO sebagai solusi permasalahan pada penelitian ini. K-NN digunakan untuk mengklasifikasikan gejala-gejala yang ada dan algoritme PSO digunakan untuk optimasi seleksi fitur dan penentuan nilai variabel *k* dalam metode K-NN.

2. METODE PENELITIAN

Dalam penelitian ini terdapat 2 proses utama seperti pada Gambar 2 yaitu normalisasi data dan PSO-KNN. Data yang digunakan dalam penelitian ini adalah dataset gejala hama jeruk yang didapatkan dari pakar Balai Besar

Penelitian Tanaman Jeruk dan Buah Subtropika, Batu. Data terdiri dari 240 *dataset* dan klasifikasi berupa 5 macam hama yaitu kutu loncat, kutu daun, kutu sisik, tungau, dan *thrips*.



Gambar 2. Alur Metodologi Penelitian

2.1 Sistem Pakar

Sistem pakar merupakan bagian dari bidang kecerdasan buatan yang memiliki pengalaman dan pengetahuan yang diperoleh dari satu atau banyak pakar pada suatu sistem yang dapat membantu setiap orang yang menggunakannya untuk menyelesaikan suatu masalah yang spesifik (Kusrini, 2006). Sistem pakar akan membantu semua orang termasuk orang awam sekalipun untuk menyelesaikan suatu masalah yang cukup susah yang sebenarnya hanya para ahli yang bisa melakukannya. Terdapat tiga komponen utama dalam penerapan sistem pakar.

1. *User Interface*, merupakan perantara penhubung antara pengguna dan sistem pakar. *Interface* mendapat informasi dari pengguna dan mengimplementasikannya dalam bentuk yang dapat dipahami oleh sistem. Kemudian *interface* menyajikannya ke bentuk yang bisa dipahami oleh pengguna.
2. *Knowledge Based*, suatu pengetahuan untuk penyelesaian masalah. Komponen ini terdiri atas dua elemen dasar yaitu aturan dan fakta. Fakta adalah informasi dari suatu obyek pada suatu permasalaha. Aturan merupakan informasi tentang bagaimana mendapatkan suatu fakta baru dari fakta yang telah diketahui sebelumnya.
3. *Inference Engine*, komponen ini adalah suatu mekanisme penalaran yang digunakan oleh para pakar untuk memecahkan masalah dengan memberikan metodologi penalaran dalam basis pengetahuan untuk memformulasikan kesimpulan.

2.2 *Case Based Reasoning*

Merupakan penalaran berbasis kasus dengan metode mempelajari kasus-kasus yang sudah ada untuk menyelesaikan permasalahan dengan kasus yang baru. Hal tersebut memungkinkan pembelajaran berkelanjutan dengan memperbarui basis pengetahuan menggunakan data baru dari masalah yang telah dipecahkan sehingga dapat digunakan untuk memecahkan masalah serupa di masa depan. Proses pembelajaran penalaran ini membutuhkan metode yang dikembangkan dengan baik untuk dapat mengekstrak pengetahuan yang relevan dari pengalaman dan untuk mengintegrasikan sebuah kasus ke dalam struktur pengetahuan yang ada (Faia, 2017). Terdapat empat tahapan pada *Case Based Reasoning (CBR).*

1. *Retrieve* untuk mencari kemiripan pola dari kasus-kasus yang sudah ada terhadap kasus yang baru.
2. *Reuse* untuk mencari solusi dengan memanfaatkan basis pengetahuan yang ada dan berdasarkan kemiripan kasus lama yang paling relevan dengan kasus baru.
3. *Revise* untuk meninjau kembali solusi yang dihasilkan dengan melakukan evaluasi hingga mendapatkan solusi yang paling akurat.
4. *Retain* untuk menyimpan data kasus bar uke dalam basis pengetahuan apabila solusi yang didapatkan sudah tepat sehingga nantinya dapat diterapkan pada kasus lain yang serupa.

2.3 Normalisasi Data

Dataset umumnya memiliki atribut yang berskala sangat berbeda sehingga saat perhitungan pada *dataset* ataupun peritungan jarak antar atribut data pada KNN akan menimbulkan ketimpangan jika ada atribut berskala lebih besar. Oleh karena itu, perlu

adanya menormalisasi semua atribut antara 0 sampai 1. Perhitungan normalisasi tersebut dapat dilakukan dengan Perasamaan 1.

$$a_i = (v_i - \min v_i)/(\max v_i - \min v_i) \quad (1)$$

Nilai min v_i adalah nilai minimal dari *dataset* pada atribut data ke-*i*, max v_i adalah nilai maksimum dari *dataset* pada atribut data ke-*i*, a_i merupakan hasil normalisasi dan v_i merupakan data atribut data ke-*i*.

Untuk atribut nominal, jarak antara dua nilai yang tidak sama adalah satu dan jarak antara dua nilai yang sama adalah nol. Sehingga tidak perlu adanya penyekalaan karena hanya bernilai satu dan nol saja (Juan, 2008).

2.4 *K-Nearest Neighbour*

K-Nearest Neighbour (KNN) adalah suatu metode *supervised* yang bertujuan mendapatkan pola baru suatu data dengan menghubungkan pola data sebelumnya dengan yang baru untuk mengklasifikasikan kemiripan data kedalam beberapa macam kelas berdasarkan atribut yang ada (Krisandi, 2013). Kemiripan data dapat lebih dari satu sehingga KNN dapat mengambil sejumlah *k* data yang paling mirip dan mengklasifikasikan berdasarkan data yang memiliki kemiripan paling banyak. Algoritme KNN telah dimanfaatkan untuk pengenalan pola dan estimasi nilai pada statistik. Jarak yang umumnya digunakan untuk KNN adalah jarak *Euclidian* yang dapat dihitung sesuai dengan persamaan 2 dimana nilai x_i merupakan data uji dan y_i merupakan data latih pada indeks ke-*i*.

$$\sqrt{\sum_{i=i}^{k}(x_i - y_i)^2} \quad (2)$$

Algoritme KNN pada dasarnya dilakukan dengan langkah berikut.

1. Menentukan nilai *k*
2. Hitung jarak antar data dengan semua data latih.
3. Sejumlah *k* data dipilih yang paling dekat dengan data masukan. Data akan diklasifikasikan kepada kelas yang memiliki jumlah kelas yang sama dengan nilai terbanyak.

2.5 *Particle Swarm Optimization* (PSO)

Particle Swarm Optimization (PSO) merupakan algoritme evolusi yang mirip dengan algoritme genetika dengan memanfaatkan fungsi *fitness* yang digunakan untuk mengevaluasi kualitas suatu solusi permasalahan. Dalam PSO setiap solusi ada dalam ruang pencarian dipandang sebagai sebagai partikel yang mana setiap partikel memiliki nilai *fitness* untuk dioptimalkan, dan memiliki kecepatan untuk perpindahan partikel (Hanafi, 2016). PSO memiliki beberapa kelebihan yaitu sedikit parameter, mudah diterapkan, konvergensi yang cepat, dan sederhana sehingga PSO banyak diterapkan pada optimasi fungsi, optimasi metode konvensional dan klasifikasi pola (Hasanuddin, 2016).

Secara garis besar prosedur PSO dapat dilakukan dalam beberapa langkah.

1. Inisialisasi kecepatan awal bernilai 0 untuk semua partikel seperti pada Persamaan 3.

   $$(V_{i,j(t)}=0) \quad (3)$$

   Vi,j merupakan kecepatan, j adalah letak partikel dan i adalah letak individu dan t adalah iterasi.
2. Inisialisasi posisi awal partikel dengan batasan sesuai range $[x_{\min,\max}]$. proses inisialisasi posisi terdapat pada Persamaan 4.

   $$x_{(t)}=x_{\min}+r(x_{\max}-x_{\min}) \quad (4)$$

   X merupakan posisi partikel dan *r* adalah nilai *random*
3. Inisialisasi *Pbest* dan *Gbest* awal dimana pada iterasi ke 0 nilai *Pbest* sama dengan posisi awal sesuai dengan Persamaan 5 dan *Gbest* merupakan *Pbest* dengan nilai *fitness* terbaik.

   $$(Pbest_{i,j(t)}=x_{i,j(t)}) \quad (5)$$

   Pbest merupakan *personal best* pada individu ke-*i* dan partikel ke-*j*. Xij merupakan posisi partikel
4. *Update* kecepatan dilakukan untuk menentukan arah perpindahan posisi partikel yang ada di populasi. Kecepatan dihitung sesuai Persamaan 6. Terdapat Batasan untuk kecepatan yang digunakan yaitu berdasarkan nilai maksimum dan minimum posisi partikel untuk menentukan batas kecepatan maksimum dan minimum yang dipengaruhi oleh interval (k) yang sebaiknya dilakukan pada proses inisialisasi. Proses update dilakukan seperti pada Persamaan 7 dan 8.

   $$v_{i,j}^{t+1} = w.v_{i,j}^{t} + c_1 r_1 (Pbest_{i,j}^{t} - x_{i,j}^{t}) + c_2 r_2 (Gbest_{g,j}^{t} - x_{i,j}^{t} \quad (6)$$

$$v_j^{max} = k\frac{(x_{j,max} - x_{j,\min})}{2}\ k \in (0,1] \quad (7)$$

$$if\ v_{ij}(t+1) > v_j^{max}\ then\ v_{ij}(t+1) = v_j^{max} \quad (8)$$

$$if\ v_{ij}(t+1) < -v_j^{max}\ then\ v_{ij}(t+1) = -v_j^{max}$$

Nilai *c1* dan *c2* adalah koefisien akselerasi, nilai *r1* dan *r2* adalah partikel random, nilai *w* adalah bobot inertia.

5. *Update* posisi dilakukan untuk menentukan posisi terbaru dari setiap partikel berdasarkan hasil *update* kecepatan sebelumnya. Setelah didapatkan nilai kecepatan maka dilanjutan dengan perhitungan *sigmoid* dari kecepatan tersebut sesuai dengan Persamaan 9 Kemudian hasil signoid yang telah didapat akan diproses lebih lanjut pada Persamaan 10 sehingga didapatkan posisi terbaru Setelah itu menentukan hasil *fitness* terbaru yang tentunya juga akan mendapat nilai *Pbest* terbaru.

$$sig(v_{i,j}^t) = \frac{1}{2+e^{-v_{i,j}^t}},\ j = 1,2,..d \quad (9)$$

$$if\ rand[0,1] > sig(v_{i,j}^t)\ then\ x_{i,j}^{t+1} = 0$$

$$if\ rand[0,1] < sig(v_{i,j}^t)\ then\ x_{i,j}^{t+1} = 1$$

$$j = 1,2,....d \quad (10)$$

6. *Update Pbest*, yaitu dengan membandingkan nilai *fitness* dari *Pbest* pada iterasi sebelumnya dengan *fitness* dari *update* Posisi. Nilai yang terbaik akan menjadi *Pbest* yang baru pada iterasi selanjutnya.

2.6 PSO-KNN

PSO-KNN merupakan perbaikan KNN untuk meningkatkan keakuratan dari hasil klasifikasi. Nilai *k* pada KNN tidak didapatkan dari percobaan tetapi didapatkan dari hasil pencarian *PSO*. Nilai *k* dikodekan dalam biner sejumlah n_1 bit dan panjang fitur dikodekan dalam n_2 bit sehingga setiap partikel terdiri sejumlah (n_1 + n_2) bit gen. Secara teori *k* dapat bernilai 1 sampai nilai data *sample*. Nilai *k* dikodekan menjadi *substring* s_1 dengan n_1 bit dan panjang fitur dikodekan menjadi *substring* s_2 dengan n_2 bits yang mana panjang total s_1 dan s_2 akan merepresentasikan partikel pada PSO. Kemudian untuk menentukan nilai *k* dapat dirumuskan untuk mencari nilai *k* (Juan, 2008) seperti pada Persamaan 11.

$$k = 1 + decimal(s_1)\ x\ \frac{a-1}{2^{n_1}-1} \quad (11)$$

Proses PSO-KNN sebagai berikut:

1. Bangkitkan Populasi PSO
   Bangkitkan populasi terdiri dari 3 proses yaitu inisialisasi kecepatan awal, inisialisasi posisi awal, serta inisialisasi *Pbest* dan *Gbest*. Inisialisasi dilakukan dengan panjang partikel dalam biner sejumlah *N* string, *N* merupakan jumlah (n_1 + n_2). Kemudian hitung *fitness* untuk mendapatkan *Gbest* seperti pada langkah ke 4.
2. Update Kecepatan PSO
3. Update Posisi PSO
4. Hitung *fitness* (evaluasi akurasi KNN)
   Hitung nilai *fitness* setiap individu pada populasi, menggunakan nilai akurasi KNN berdasarkan rata-rata dari nilai klasifikasi yang benar atau sesuai dengan data asli dan berdasarkan nilai *k* dari setiap individu.
5. *Update Pbest* PSO
6. *Termination test*
   Ulangi langkah 2 sampai 5 hingga kondisi berhenti terpenuhi.

2.7 Evaluasi

Evaluasi dilakukan untuk mengetahui tingkat keberhasilan dari klasifikasi yang dihasilkan oleh sistem. Tingkat keberhasilan ini berdasarkan hasil klasifikasi yang relevan dengan data asli maupun yang tidak relevan atau tidak sesuai dengan data asli. Konsep evaluasi yang digunakan pada penelitian ini adalah nilai akurasi. Akurasi didapatkan dari perbandingan terhadap jumlah data yang relevan atau sesuai dengan data asli dengan seluruh jumlah data yang ada dan kemudian dikonversikan ke bentuk persentase sesuai pada persamaan 12.

$$Akurasi = \frac{\sum \text{data yang relevan}}{\sum \text{data testing}} \times 100\% \quad (12)$$

3. PENGUJIAN & ANALISIS HASIL

Pada bab ini akan membahas hasil pengujian dari Optimasi KNN Menggunakan KNN pada Sistem Pakar Untuk Monitoring pengendalian hama pada tanaman Jeruk. Pengujian dilakukan dengan beberapa uji coba agar menemukan parameter yang teroptimal untuk penelitian ini. Pengujian program berupa:

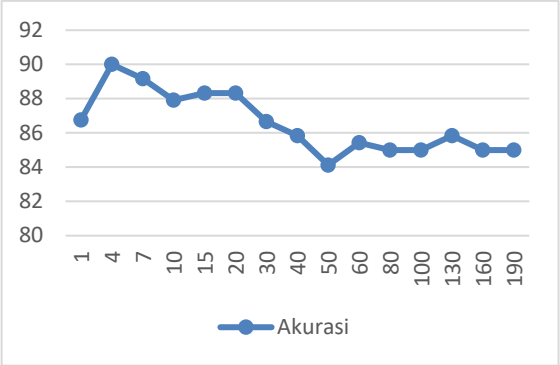
1. Uji coba pengaruh *k* pada KNN sebelum dioptimasi.

2. Uji coba akurasi KNN menggunakan *k-Fold cross validation*
3. Uji coba pengaruh iterasi PSO terhadap akurasi.
4. Uji coba pengaruh *popsize* (jumlah partikel) PSO terhadap akurasi
5. Uji coba pengaruh *c1* dan *c2* (koefisien akselersi) PSO terhadap akurasi
6. Uji coba pengaruh *w* (bobot inertia) PSO terhadap akurasi

Pengujian dilakukan berdasarkan nilai akurasi yang dengan membandingkan nilai hasil klasifikasi dan nilai sesungguhnya. PSO-KNN merupakan KNN yang nilainya telah teroptimasi oleh PSO. Sehingga penelitian ini akan melakukan pencarian nilai rata-rata akurasi dari setiap uji coba yang dilakukan guna mengetahui parameter-parameter terbaik dari PSO-KNN pada penerapan penelitian ini.

3.1 Hasil dan Analisis Uji Coba Pengaruh Parameter *K* Terhadap Akurasi Pada KNN

Uji coba ini bertujuan untuk mengetahui pengaruh banyaknya tetangga atau parameter *k* pada KNN terhadap akurasi dan mengetahui nilai *k* terbaik sebagai tolak ukur sebelum dioptimasi menggunakan PSO yang akan diterapkan pada penelitian ini. Uji coba ini dilakukan dengan membagi dataset menjadi 2 bagian yaitu 48 data uji dan 192 data latih. Kemudian Sistem dijalankan dengan nilai *k* yang berbeda-beda. Pada penelitian ini, percobaan dilakukan pada 15 nilai *k* yang berbeda. Pada setiap percobaan akan diketahui nilai akurasi yang dihasilkan pada setiap *k*. Hasil klasifikasi didapatkan dari perbandingan hasil sistem dengan data pakar. Hasil yang sesuai dengan data pakar akan dianggap sebagai data benar, yang kemudian akan dilakukan perhitungan rata-rata data benar dari total data keseluruhan dan dikonversi ke bentuk presentase.

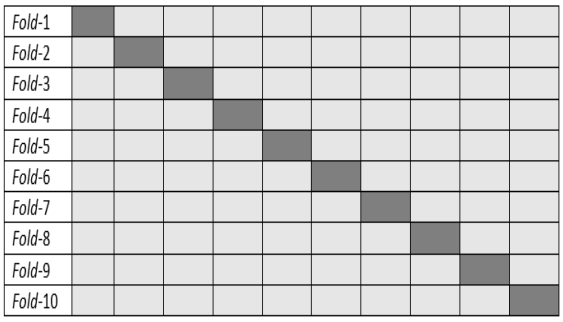


Gambar 3 Hasil Uji Coba Nilai *k*

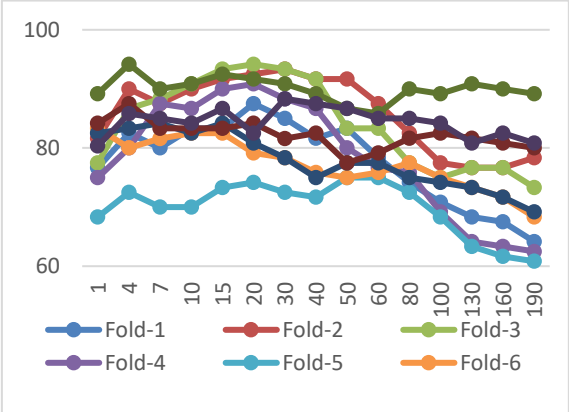
Sesuai dengan grafik pengujian pada Gambar 3. Nilai akurasi yang dihasilkan pada tiap *k* berbeda-beda. Dari percobaan tersebut diperoleh nilai akurasi tertinggi sebesar 90 % pada *k* = 4 dan terendah sebesar 84,12 % pada *k* = 50. Grafik menunjukkan bahwa semakin tinggi nilai *k* maka akurasi yang dihasilkan akan cenderung menurun akan tetapi dari percobaan awal hingga akhir tidak mengalami perubahan yang signifikan tetapi masih cukup bagus dikarenakan kedekatan karakteristik dari data latih dan data uji yang digunakan. Namun pada penelitian ini bentuk grafik terlihat naik turun tidak beraturan. Hal ini mengindikasikan bahwa terjadi *overfitting* pada data yang digunakan. *overfitting* disebabkan karena sebaran data latih dan data uji yang tidak tidak ideal dan terdapat semacam *noise*. Hal ini disebabkan karena menggunakan data yang sudah terdaftar sebelumnya tanpa mengatur kembali sebaran data tersebut. Selain itu permasalahan ini juga disebabkan karena kompleksitas data latih yang besar (Walega, 2014).

3.2 Hasil dan Analisis Uji Coba Akurasi KNN Menggunakan *K-Fold Cross Validation*

Uji coba ini bertujuan untuk mengetahui nilai akurasi pada KNN dengan menggunakan metode *k-Fold cross validation*. Pada percobaan ini, peneliti menggunakan 10-*fold* yang berarti data terbagi menjadi 10 subset yang terdiri dari 24 data tiap subset. Setiap subset akan berkesempatan menjadi data uji sebanyak 1 kali dan selebihnya akan menjadi data latih pada 10 percobaan. Proses dari pembagian *fold* pada metode ini akan ditunjukkan pada Gambar 4. Subset berwarna gelap merupakan subset yang digunakan sebagai data uji sedangkan yang berwarna lebih terang akan menjadi data latih.



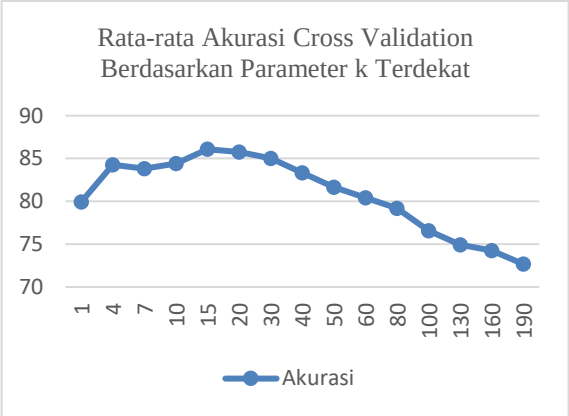
Gambar 4 Pembagian Subset dan Proses 10-*Fold cross validation*



Gambar 5 Grafik Uji Coba Akurasi Cross Validation pada Setiap *Fold*

Nilai akurasi yang dihasilkan pada setiap *fold* dengan beberapa parameter *k* (banyaknya tetangga) hasilnya berbeda-beda. Grafik pada Gambar 6. menunjukkan bahwa akurasi tertinggi dihasilkan pada *fold*-9 sedangkan yang terendah terdapat pada *fold*-5. Perbedaan akurasi pada setiap *fold* disebabkan karena karakteristik data uji dan data latih pada setiap *fold* berbeda-beda. Pada *fold*-9 menghasilkan akurasi tinggi karena pada data uji dan data latih memiliki kedekatan karakteristik sedangkan pada *fold*-5 antara data latih dan data uji memiliki karakteristik yang cukup jauh. Grafik yang cenderung naik turun pada setiap *fold* semakin memperkuat analisis pada bab 3.1 yaitu terjadinya *overfitting* pada dataset yang digunakan dalam penelitian ini.

Kemudian rata-rata akurasi setiap *fold* menunjukkan akurasi tertinggi diperoleh pada nilai *k* = 15 yaitu sebesar 86,08% dan yang terendah pada percobaan terakhir dengan nilai *k* = 190 sebesar 72,67%. Hal tersebut bisa disimpulkan bahwa semakin tinggi nilai *k* maka akurasi yang dihasilkan akan semakin menurun. Kemudian jika dibandingkan, grafik akurasi pada Gambar 6 menunjukkan bahwa grafik lebih seimbang jika dibandingkan dengan Gambar 5. Hal ini membuktikan bahwa metode cross validation dapat digunakan untuk mengatasi terjadinya *overfitting* (Leng, 2013). Namun perbandingan akurasi masih lebih baik pada Gambar 5. Hal ini dikarenakan pada pembagian subset dari penerapan cross validation dalam penelitian ini *fold* yang memiliki karakterististik yang dekat antara data latih dan data uji lebih sedikit dari pada *fold* yang memiliki karakteristik data yang jauh sehingga rata-rata akurasi yang dihasilkan cenderung kurang optimal.



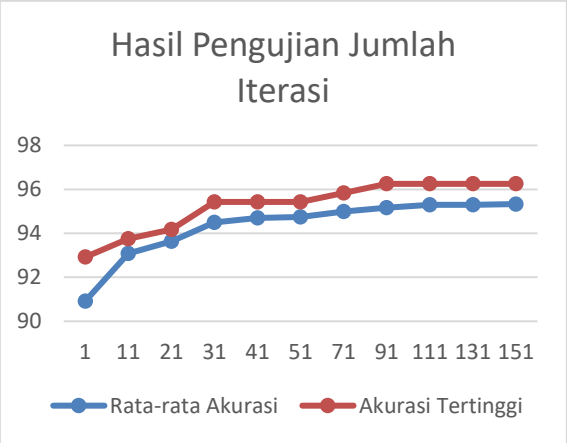
Gambar 6 Grafik Uji Coba Rata-rata Akurasi Cross Validation

3.3 Hasil Dan Analisis Uji Coba Pengaruh Jumlah Iterasi.

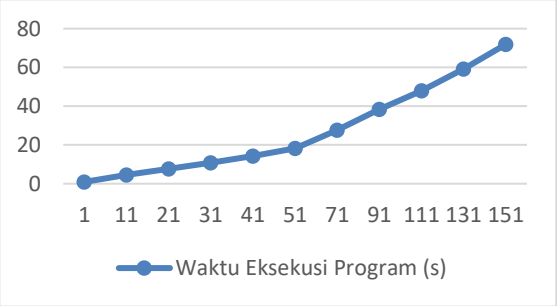
Uji coba ini bertujuan untuk mengetahui berapa pengaruh jumlah iterasi yang digunakan untuk sistem agar menghasilkan hasil yang optimal berupa nilai akurasi klasifikasi KNN yang lebih baik. Uji coba ini dilakukan dengan cara menjalankan sistem dengan nilai iterasi yang berbeda-beda, sedangkan parameter lainnya memiliki nilai sebagai berikut :

popsize : 5
c1 : 1
c2 : 1
w : 0,5

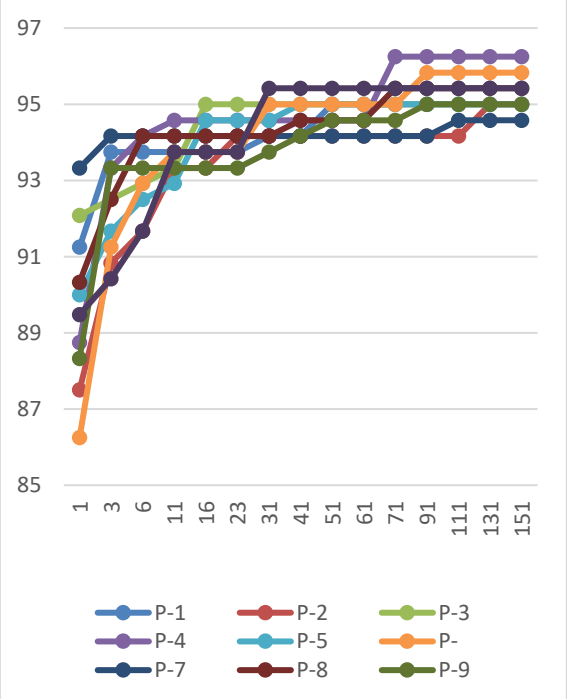
Dapat dilihat bahwa dari uji coba jumlah iterasi PSO berhasil diperoleh rata-rata akurasi secara tertinggi sebesar 95,33% dan nilai akurasi tertinggi sebesar 96,25% pada iterasi ke 151. Dari hasil ini dapat disimpulkan bahwa semakin tinggi jumlah iterasi maka akurasi yang dihasilkan akan semakin tinggi. Hal ini dibuktikan sesuai grafik pada Gambar 7 dan Gambar 9 yang hasilnya cenderung mengalami peningkatan pada jumlah iterasi yang semakin tinggi.



Gambar 7 Hasil Uji Coba Iterasi PSO



Gambar 8 Waktu Eksekusi Uji Coba Iterasi



Gambar 9 Hasil Uji Coba Konvergensi

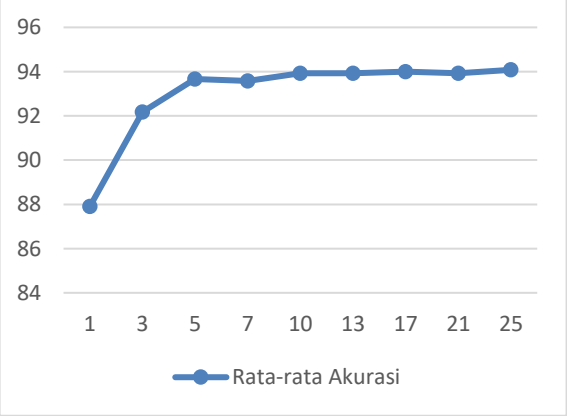
Namun semakin tinggi jumlah iterasi maka waktu yang dibutuhkan untuk program dalam melakukan komputasi juga semakin lama sehingga performa akan menurun. Sebaliknya jika iterasi sedikit waktu komputasi akan lebih cepat tetapi *fitness* atau akurasi yang dihasilkan belum tentu optimal.

Kemudian perbedaan akurasi yang berbeda pada beberapa percobaan pada iterasi yang sama menunjukkan bahwa inisialisasi partikel memberikan pengaruh terhadap proses optimasi. Hal ini dikarenakan karena inisialisasi partikelnya dilakukan secara acak sehingga mempengaruhi susunan partikel dan *fitness* yang dihasilkan pada setiap partikel. Besaran nilai *fitness* yang dihasilkan itulah yang akan mempengaruhi hasil optimasi karena menentukan waktu untuk menjelajah ruang solusi.

3.4 Hasil Dan Analisis Uji Coba Pengaruh *Popsize* (Jumlah Partikel)

Uji coba ini bertujuan untuk mengetahui berapa jumlah *popsize* yang optimal untuk digunakan pada sistem agar menghasilkan hasil yang berupa nilai akurasi klasifikasi KNN yang lebih baik. Uji coba ini dilakukan dengan cara menjalankan sistem dengan jumlah *popsize* yang berbeda-beda, sedangkan parameter lainnya memiliki nilai sebagai berikut :

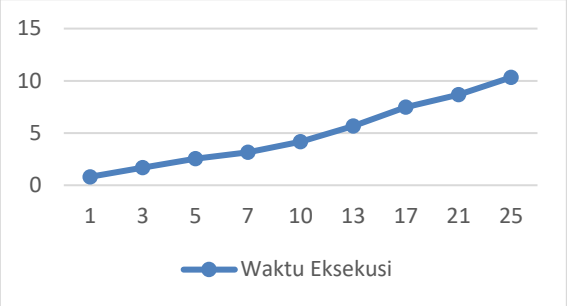
iterasi : 5
c1 : 1
c1 : 1
k : 0,5
w : 0,5



Gambar 10 Hasil Uji Coba *popsize*

Dapat dilihat bahwa dari uji coba jumlah *popsize* PSO berhasil diperoleh rata-rata akurasi tertinggi sebesar 94,08 % pada *popsize* sebanyak 25. Dari hasil ini dapat disimpulkan bahwa semakin tinggi jumlah *popsize* maka akurasi yang dihasilkan akan semakin tinggi. Hal ini dibuktikan sesuai grafik pada Gambar 9 yang hasilnya cenderung mengalami peningkatan pada jumlah *popsize* yang semakin tinggi

meskipun ada beberapa yang sedikir menurun karena inisialisasi bernilai *random*. Grafik yang meningkat ini disebabkan dengan semakin banyaknya *popsize* maka semakin luas ruang solusi yang dapat dijelajahi oleh partikel seiring dengan semakin banyaknya jumlah partikel. Namun semakin banyaknya jumlah partikel yang dijelajahi jelas maka waktu yang dibutuhkan untuk program dalam melakukan komputasi juga semakin lama sehingga performa akan menurun.



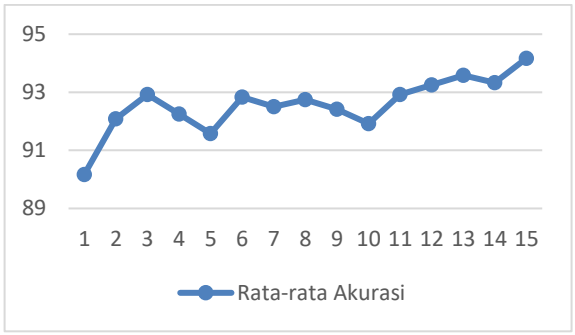
Gambar 11 Waktu Eksekusi Uji Coba *Popsize*

Kemudian apabila *popsize* terlalu sedikit maka semakin rendah pula kemungkinan pastikel menghasilkan *fitness* terbaik. Hal ini disebabkan *popsize* yang sedikit sangat dipengaruhi proses inisialisasi partikel. Apabila inisialisasi menghasilkan partikel dengan *fitness* yang baik dan konsisten dalam beberapa percobaan maka tanpa membutuhkan banyak iterasi akan tetap mampu mendapat *Gbest* yang cukup baik. Namun jika terjadi sebaliknya, yaitu inisialisasi menghasilkan *fitness* rendah maka besar kemungkinan akan mendapatkan *Gbest* yang juga rendah. Hal tersebut disebabkan karena semakin sedikit *popsize* yang terlibat pada iterasi yang sama maka ruang solusi yang dijelajahi akan semakin sedikit juga.

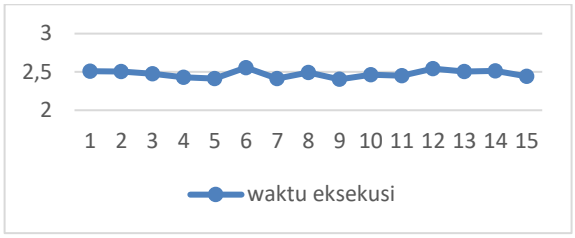
3.5 Hasil dan Analisis Uji Coba Pengaruh *c1* dan *c2 (Koefisien Akselerasi)*

Uji coba ini bertujuan untuk mengetahui berapa jumlah *popsize* yang optimal untuk digunakan pada sistem agar menghasilkan hasil yang berupa nilai akurasi klasifikasi KNN yang lebih baik. Uji coba ini dilakukan dengan cara menjalankan sistem dengan jumlah *popsize* yang berbeda-beda, sedangkan parameter lainnya memiliki nilai sebagai berikut :

iterasi : 5
popsize : 5
w : 0,5



Gambar 12 Hasil Uji Coba *c1* dan *c2*



Gambar 13 Waktu Eksekusi Uji Coba *c1 & c2*

Dapat dilihat bahwa dari uji coba kombinasi *c1* dan *c2* berhasil diperoleh rata-rata akurasi tertinggi sebesar 94,17% pada nilai *c1*=1 dan *c2* = 1,2. Dari grafik pada Gambar 10 menunjukkan bahwa apabila *c1* lebih besar daripada *c2* maka penentuan kecepatan nilai *Pbest* akan lebih dominan, sehingga mengakibatkan partikel dengan nilai *fitness* yang rendah hanya akan berputar-putar di dalam *Pbest* atau tidak bergerak menuju *Gbest*. Sehingga partikel dengan nilai yang rendah akan butuh lebih banyak iterasi agar nilai *fitness*nya dapat menigkat dengan pesat.

Sebaliknya, apabila *c2* yang nilainya lebih besar daripada *c1* maka *Gbest* akan sangat berpengaruh. Ini juga bukan hal bagus karena dapat mengakibatkan partikel yang memiliki nilai *fitness* yang lebih rendah bisa akan bergerak sangat cepat sehingga berpindah terlalu cepat yang mana pada kondisi tersebut ada kemungkinan partikel akan melewati bagian ruang solusi yang memiliki *fitness* terbaik. Hal itu kemungkinannya akan lebih besar jika nilai *Gbest* yang ada terlalu bagus (juneja, 2016).

Agar bisa memperoleh *fitness* yang lebih baik maka dibutuhkan nilai *c1* dan *c2* yang tepat. Dilihat dari hasil uji coba ini bisa dikatakan jika nilai *c2* lebih besar dari *c1* maka akan menghasilkan *fitness* yang lebih baik dan stabil dibandingkan *c2* yang nilainya mendekati 0 dan dibawah *c1*.

Hal ini ditunjukkan dengan hasil uji coba yang mana nilai *c2* yang lebih besar dari *c1* tapi dalam interval yang dekat hasilnya lebih optimal

daripada *c2* yang lebih besar dari *c1* namun intervalnya terlalu jauh. Sehingga ini bisa menjadi bukti ji partikel akan lebih baik jika partikel sedikit lebih condong bergerak menuju *global best* dari pada *local best.*

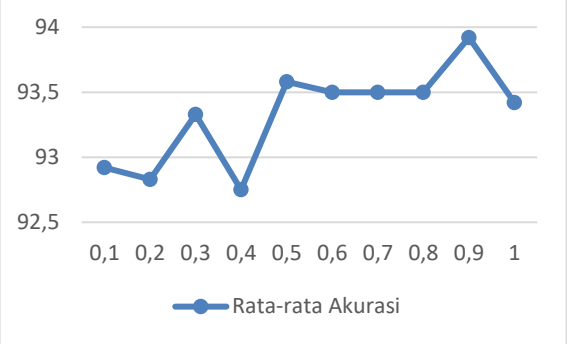
3.6 Hasil dan Analisis Uji Coba Pengaruh *W* (Bobot Inertia)

Uji coba ini bertujuan untuk mengetahui berapa jumlah *popsize* yang optimal untuk digunakan pada sistem agar menghasilkan hasil yang berupa nilai akurasi klasifikasi KNN yang lebih baik. Uji coba ini dilakukan dengan cara menjalankan sistem dengan jumlah *popsize* yang berbeda-beda, sedangkan parameter lainnya memiliki nilai sebagai berikut :

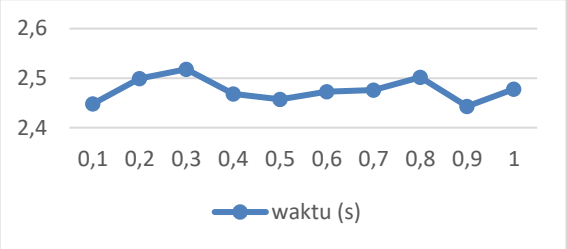
popsize : 5
iterasi : 5
c1 : 1
c2 : 1

Dapat dilihat bahwa dari uji coba jumlah *popsize* PSO berhasil diperoleh rata-rata akurasi tertinggi sebesar 93,92% pada nilai *w* = 0,9. Dari hasil ini dapat disimpulkan bahwa semakin tinggi nilai *w* maka kemungkinan mendapatkan *fitness* yang lebih baik semakin besar. Hasil ini juga sesuai dengan penelitian (juneja, 2016) yang mana nilai *w* terbaik adalah diatas 0,5 dan kurang dari 1. Hal ini disebabkan karena apabila nilai *w* lebih dari 1 maka dapat berakibat partikel di dalam PSO menjadi tidak stabil karena kecepatan yang dihasilkan tidak terkontrol. Terbukti dari hasil uji menunjukkan pada rentang nilai tersebut memang menghasilkan rata-rata akurasi yang lebih baik dan mulai menurun lagi ketika memasuki niali *w* = 1.

Kemudian pada hasil uji juga menujukkan bahwa nilai *w* yang kecil dapat mengakibatkan peran kecepatan parikel tidak berpengaruh dan dapat meningkatkan peluan terjadinya konvergensi dini. Kemudian apabila kecepatan tidak terlalu berpengaruh maka proses perpindahan partikel saat update posisi akan melambat. Sehingga hal tersebut mengakibatkan partikel yang nilai *fitness*nya memiliki perbedaan signifikan dengan nilai *fitness* terbaik akan membutuhkan iterasi lebih banyak untuk bisa mendekati posisi dari partikel yang memiliki *fitness* terbaik (Clerc, 2011).



Gambar 14 Hasil Uji Coba *w*



Gambar 15 Waktu Eksekusi Uji Coba *w*

3.7 Hasil Pengujian dan Analisis Global

Pada beberapa uji coba yang telah dilakukan didapatkan beberapa kesimpulan. Pada pengujian akurasi KNN, diketahui bahwa nilai *k* dan sebaran data sangat mempengaruhi akurasi yang dihasilkan. Kemudian dari hasil percobaan *K-Fold cross validation* dapat diketahui bahwa meskipun metode validasi tersebut mampu untuk mengatasi sebaran data yang tidak ideal namun hal itu tidak menjamin akurasi yang dihasilkan semakin tinggi. Dengan demikian pada permasalahan ini memang dibutuhkan solusi untuk mengoptimalkan nilai *k* dan sebaran data yang tidak ideal yaitu dengan optimasi dan seleksi fitur menggunakan PSO.

Dalam hasil uji coba parameter PSO, jumlah iterasi dan *popsize* sangat mempengaruhi nilai fitness yang dihasilkan. Hal ini disebabkan banyaknya iterasi dan *popsize* akan berdampak terhadap daya jelajah partikel yang akan megalami proses perpindahan partikel yang semakin banyak seiring dengan bertambahnya waktu dan luas ruang solusi yang dihasilkan oleh iterasi dan *popsize*. Namun, meskipun nilai fitness yang dihasilkan lebih optimal, hal tersebut sangat mempengaruhi performa karena semakin banyak iterasi dan *popsize* akan membutuhkan waktu komputasi yang lebih lama. Sehingga jumlah iterasi maksimum akan lebih baik berdasarkan kompleksitas dari suatu permasalahan yang ada. Kemudian pada uji parameter w, *c1* dn *c2* dapat diketahui bahwa

ketiganya berpengaruh pada kecepatan partikel saat melakukan perpindahan posisi. Nilai *w* menjadi kunci untuk mengatur seberapa cepat partikel bergerak yang mana kecepatan tidak boleh terlalu lambat atau cepat agar dapat menemukan hasil fitness yang paling optimal. Sedangkan untuk *c1* dan *c2* berpengaruh terhadap arah gerak suatu partikel apakah menuju *local best* ataupun *global best*. Untuk mendapatkan hasil yang optimal maka nilai kedua parameter tersebut setidaknya mendekati seimbang dan tidak dominan salah satunya sehingga partikel dapat bergerak dalam porsi yang tepat untuk mendapatkan solusi terbaik.

Kemudian pada seluruh uji coba parameter, PSO mampu menghasilkan *fitness* atau akurasi yang lebih baik daripada akurasi yang dihasilkan KNN sebelum dioptimasi. Hal tersebut menunjukkan bahwa PSO dapat mengatasi beberapa kekurangan KNN dalam penelitian ini dengan cara mencari fitur yang tepat melalui penjelajahan ruang solusi yang bervariasi sehingga PSO dapat melakukan evaluasi secara berkala untuk memperbaiki hasil sebelumnya.

4 KESIMPULAN

Dari hasil pengujian dan pembahasan pada bab-bab sebelumnya maka didapatkan kesimpulan sebagai berikut:

1. Implementasi KNN pada penelitian ini dapat diterapkan dengan baik untuk melakukan klasifikasi pada 5 label klasifikasi hama. Proses yang dilakukan KNN adalah dengan menghitung jarak Euclidian dari data yang telah dinormalisasi dan kemudian dihitung tetangga terdekatnya. Dari hasil pengujian dapat disimpulkan bahwa nilai *k* memiliki pengaruh terhadap akurasi KNN dan mencapai akurasi tertinggi sebesar 90% pada *k* sebesar 4 dan pada pengujian *k-Fold cross validation* menghasilkan rata-rata akurasi tertinggi sebesar 86,08% pada *k* sebesar 15. Hasil akurasi KNN menggunakan *k-Fold cross validation* lebih rendah dikarenakan sebagian besar komposisi pembagian *fold* memiliki karakteristik yang tidak berdekatan kurang merata namun lebih seimbang sehingga mampu mengatasi masalah *overfitting* pada KNN biasa.
2. Implementasi algoritme PSO dapat diterapkan dengan baik dalam proses optimasi KNN pada sistem pakar untuk Monitoring Pengendalian Hama pada tanaman Jeruk. Algoritme PSO berperan dalam proses optimasi nilai *k* dan seleksi fitur pada KNN.
3. Dalam penerapan PSO-KNN, akurasi KNN yang masih belum maksimal dapat teratasi dengan optimasi menggunakan PSO. Hal tersebut dipengaruhi oleh seleksi fitur dan optimasi nilai *k* yang dihasilkan oleh PSO. Hasilnya akan lebih optimal jika menggunakan parameter yang sudah teruji. Dari hasil pengujian yang telah berhasil diperoleh paramaeter PSO yang terbaik yaitu *iterasi* sejumlah 131, *popsize* sebanyak 25, nilai *c1* sebesar 1, nilai *c2* sebesar 1,2 dan nilai *w* sebesar 0,9. Terbukti pada percobaan PSO didapat akurasi tertinggi sebesar 96,25%. Akurasi tersebut mampi meningkatkan akurasi KNN yang hanya mencapai 90%. Dapat disimpulkan PSO dapat mengatasi beberapa kekurangan KNN dalam penelitian ini dengan cara mencari fitur yang tepat melalui penjelajahan ruang solusi yang bervariasi dengan proses evaluasi secara berkala untuk memperbaiki hasil yang sudah ada sebelumnya.

DAFTAR PUSTAKA

Badan Pusat Statistik. 2015. *Produksi Buah-buahan Nasional Tahun 2006-2015*. <http://www.bps.go.id> [Diakses pada tanggal 7 Maret 2017].

Clerc, M. 2011. *Particle Swarm Optimization ISTE*. New York : United States of America.

Endarto, O., Martini, E. 2016. *Budi Daya Jeruk Sehat*. Batu : Balitjesto.

Faia, R., Pinto, T., Abrishambaf, O., Fernandes, F., Vale, Z. 2017. *Case Based Reasoning With Expert System and Swarm Intelligence to Determine Energy Reduction in Buildings Energy Management.* Spain : University of Salamanca.

Hanafi., Cabrera, F.M., Dimane, F., Manzanares, J.T. 2015. *Application Of Particle Swarm Optimization For Optimizing The Process Parameters In Turning Of Peek CF30 Composites.* Romania : Tirgu-Mures.

Hasanuddin. 2016. *Perbandingan Algoritma K-NN Dan K-NN-PSO Untuk Klasifikasi Tingkat Pengetahuan Ibu Dalam*

Pemberian Asi Eksklusif. Uniska.

Ichsan, M.C., Prayuginingsih, H. 2016. *Pengembangan Model Peningkatan Daya Saing Jeruk Lokal Untuk Memperkokoh Ekonomi Masyarakat Pedesaan.* Jember : Universitas Muhammadyah Jember.

Juan, Zhang, Niu Yi, He Wenbin. 2008. *Using Genetic Algorithm to Improve Fuzzy k-NN*. Dongguan: Dongguan University of Technology.

Juneja, M., Nagar, S.K. 2016. *Particle Swarm Optimization Algorithm and its Parameter: A review.* Internation Conference on Control Computing, Communication and Materials.

Krisandi, N., Helmi, Prihandono, B. 2013. *Algoritma K-Nearest Neighbor Dalam Klasifikasi Data Hasil Produksi Kelapa Sawit Pada PT. Minamas Kecamatan Parindu*. Pontianak : Untan.

Kusrini. 2006. *Sistem Pakar Teori dan Aplikasi*. Yogyakarta : Andi.

Leng, Z., Gao, J., Qin, Y., Liu, X., & Yin, J., 2013. *Short-term Forecasting Model of Traffic Flow Based on GRNN*. *Chinese Control and Decision Conference*.

Mutrofin, S., Izzah, A., Kurniawardhani, A., Masrur, M. 2014. *Optimasi Teknik Klasifikasi Modified k Nearest Neighbor Menggunakan Algoritme Genetika.* Jombang: Unipdu.

Naga. 2011. *Sentra Jeruk di Subang diserang Hama.* < http://news.karirsumut.com/view/news/339404/sentra_jeruk_di_subang_diserang_hama.html > [Diakses pada tanggal 24 April 2017].

Priadi, R. 2015. *Petani Jeruk di Karo Beralih ke Kopi.* < http://www.antarasumut.com/berita/154650/petani-jeruk-di-karo-beralih-ke-kopi. > [Diakses pada tanggal 2 Mei 2017].

Walega, P.A. 2014. *Overfitting Problem In A Virtual Sensor Obtained With W-M Method.* Warsawa. : Institue of philosophy.

Zyout, I.**,** Abdel-Qader, I. 2011. *Classification Of Microcalcification Clusters Via PSO-K-NN Heuristic Parameter Selection And GLCM Features.* Jordan : Tafila Tecnical University.