REPOSITOR, Vol. 3, No. 1, Januari 2021, Pp. 97-110
ISSN : 2714-7975
E-ISSN : 2716-1382

Optimisasi Algoritma K-Means Menggunakan Artificial Bee Colony pada Content-Based Image Retrieval

Agus Eko Minarno[*1], Yufis Azhar[2], Yudhono Witanto[3]
[1,2,3]Universitas Muhammadiyah Malang
agoes.minarno@gmail.com[*1], yufis.az@gmail.com[2], yudhonowitanto.yw@gmail.com[3]

Abstrak

Content-based Image Retrieval (CBIR) merupakan implementasi dari teknik computer vision pada kasus image retrieval yang merupakan kasus pencarian gambar digital pada database yang sangat besar. Pada penelitian ini diperkenalkan metode clustering baru untuk sistem CBIR, metode yang digunakan merupakan kombinasi antara algoritma Artificial Bee Colony (ABC) dengan K-Means. Tiga fitur digunakan untuk mengekstraksi fitur – fitur yang dimiliki gambar, yaitu: RGB Color Feature, Edge Feature, dan Grey Level Co-occurrence Matrix (GLCM). Ketiga fitur tersebut tergabung dalam algoritma Multi Texton Co-occurrence Descriptor (MTCD). K-means merupakan algoritma pengelompokkan yang banyak digunakan pada kasus clustering suatu data, K-means banyak digunakan karena implementasinya yang sangat mudah. Meskipun demikian, algoritma ini memiliki kelemahan, salah satunya yakni dalam penentuan titik awal centroid. Artificial Bee Colony (ABC) merupakan algoritma optimasi yang cara kerjanya mengadopsi dari cara koloni lebah dalam mencari makanan. Metode ABC diketahui dapat memecahkan permasalahan local optimum, yang pada umumnya terjadi pada penggunaan K-means karena kelemahannya dalam menentukan centroid. Metode yang akan digunakan pada penelitian ini merupakan kombinasi dari algoritma ABC dengan K-means untuk mengelompokkan fitur – fitur yang telah diekstraksi yang diimplementasikan pada dataset Corel-10.000 dan Batik. Kombinasi dari kedua algoritma ini dapat menjadi solusi dalam permasalahan pengelompokkan data atau clustering di ranah data science. Penelitian ini bertujuan untuk mengetahui seberapa efektif dan efisien penggunaan kombinasi algoritma ABC dan K-means dalam clustering pada fitur dataset Corel-10.000 dan Batik.

Kata Kunci: *Image retrieval, Clustering, MTCD, Artificial Bee Colony, K-means*

Abstract

Content-based Image Retrieval (CBIR) is an implementation from computer vision techniques to the image retrieval problem, that is the problem for searching digital images in large databases. This research proposed a new method for CBIR system, combining Artificial Bee Colony and K-means algorithm. Three features are used to retrieve the images. These features are extracted by MTCD algorithm which included RGB Color Feature, Edge Feature, and Grey Level Co-occurrence Matrix (GLCM). K-means is a clustering algorithm that used in many clustering problems, this algorithm is widely used because it is fluently works in every cases. However, this algorithm has weaknesses, one of which is in the initialization of centroid. Artificial Bee Colony (ABC) is an optimization algorithm that works by adopting bee colony behavior in finding food. This method is known to solve local optimum problems, which generally occur in the use of K-means because of its weakness in determining centroids. This research proposed a new method in clustering that combining ABC and K-means to cluster features that extracted from Corel-10.000 and Batik dataset. Combining these two algorithms can be a solution for data clustering in field of data science. The goal of this research is to determine how effective and efficient the combination of ABC and K-means algorithm in clustering the Corel-10.000 and Batik dataset.

Keywords: *Image retrieval, Clustering, MTCD, Artificial Bee Colony, K-means*

1. Pendahuluan

Image retrieval merupakan topik penting dalam ranah kecerdasan buatan dan pengenalan pola dewasa ini. Secara umum terdapat 3 kategori untuk metode untuk *image retrieval*, yaitu: *text-based, content-based,* dan *semantic-based.* Metode *text-based* diperkenalkan di tahun 1970,

di mana metode ini memerlukan tenaga manusia untuk memberikan anotasi teks pada setiap gambar secara manual, keakurasian anotasi pun tergantung pada persepsi.manusia.terhadap gambar, tentunya metode seperti ini sangat tidak efisien [1]. Pada tahun 1990 an awal, para peneliti banyak membangun sistem *content-based. image..retrieval*, seperti: *QIBC, MARS, Virage, Photobook, FIDS, Web Seek, Netra, Cortina* [2], *VisualSEEK* [3], dan *SIMPLIcity* [4]. kebanyakan dari sistem tersebut menggunakan fitur warna, tekstur, dan bantuk untuk diekstak yang kemudian akan dihitung kesamaannya dengan *image query*. Fitur – fitur tersebut digunakan baik secara independen maupun dikolaborasikan antara satu fitur dengan fitur yang lain.

Pada *Content-Based Image Retrieval* fitur – fitur gambar diekstrak untuk membentuk sebuah *feature vector* dimana nantinya *feature vector* ini akan diproses menggunakan berbagai macam algoritma agar menghasilkan sistem CBIR dengan tingkat akurasi dan presisi yang tinggi. Untuk mencapai tujuan tersebut beberapa penelitian sebelumnya telah melakukan penelitian baik pada metode ekstraksi fitur maupun pemrosesan *feature vector.* Metode ektraksi fitur pada penelitian [5] mengajukan metode yang bernama MTCD (*Multi Texton Co-Occurrence Descriptor*) MTCD mengekstrak beberapa fitur pada gambar secara urut menggunakan *texton*. Fitur – fitur yang diekstraksi diantaranya: warna, *texture*, dan *shape* (bentuk), setelah fitur terekstrak, selanjutnya adalah menghitung hasil dari ekstraksi fitur gambar secara menyeluruh menggunakan *Gray Level Co-occurrence Matrix* (GLCM). Data pada percobaan menyebutkan bahwa, dengan menambahkan 2 texton dan GLCM, angka *precision* naik sebesar 2.86% untuk dataset batik, 3.40% untuk dataset corel 5000, dan 3.06% untuk dataset corel 10.000.

Untuk meningkatkan nilai *precision* para peneliti banyak yang fokus meniliti pada pemrosesan *feature vector*. Penilitian yang berjudul *Image Indexing using Color Histogram and k-means Clustering for Optimization CBIR in Image Database* mengajukan metode untuk mengcluster *feature vector* Hasil percobaan menunjukkan bahwa, dengan nilai K = 5 menghasilkan nilai *precision* sebesar 0.92 [6].

Algoritma.*Artificial.Bee.Colony.*pertama.kali diperkenalkan oleh Dervis Karaboga pada tahun 2005. Algoritma ini mensimulasikan kawanan lebah yang sedang mencari..sumber. makanan [7]. Kemudian penelitian yang berjudul *A novel clustering approach: Artificial Bee Colony (ABC) algorithm* menunjukkan metode klusterisasi baru yaitu dengan menggunakan algoritma ABC. Pada..penelitian..ini algoritma ABC digunakan untuk mengklusterisasi *benchmark problem*. Hasil percobaan menunjukkan bahwa algoritma ABC mengungguli algoritma PSO lain dalam 12 masalah. Selain itu, persentase kesalahan klasifikasi rata-rata untuk semua masalah adalah 13,13% untuk ABC dan 15,99% untuk PSO [8].

Multi Texton Co-Occurrence Descriptor (MTCD) adalah salah satu metode untuk merepresentasikan fitur gambar secara global yang menggunakan data Batik dan Corel. Namun dalam metode tersebut masih dijumpai beberapa kekurangan, seperti: penambahan 2 jenis *texton* belum mampu menambah secara signifikan nilai *precision*.

Selain itu penggunaan algoritma K-means dalam sistem CBIR memiliki beberapa keuntungan. Hasil penelitian Amory menjelaskan bahwa, Membagi gambar dalam satu jendela besar tidak akan meningkatkan akurasi, sebagai gantinya kita dapat melakukan segmentasi gambar tersebut dalam jumlah kluster K yang dapat diterima [9].

Algoritma optimisasi *Artificial Bee Colony* juga memiliki keuntungan dalam hal klusterisasi dan klasifikasi. Hal ini dijelaskan pada hasil penelitian karaboga di tahun 2011 yang menyatakan bahwa, *test error rates* dan peringkat pada tabel hasil percobaan menunjukkan bahwa klustering menggunakan algoritma ABC menghasilkan kapabilitas generalisasi yang luar biasa. Dengan hasil tersebut dapat di klaim bahwa algoritma ABC dapat digunakan dalam kasus klustering maupun klasifikasi [8].

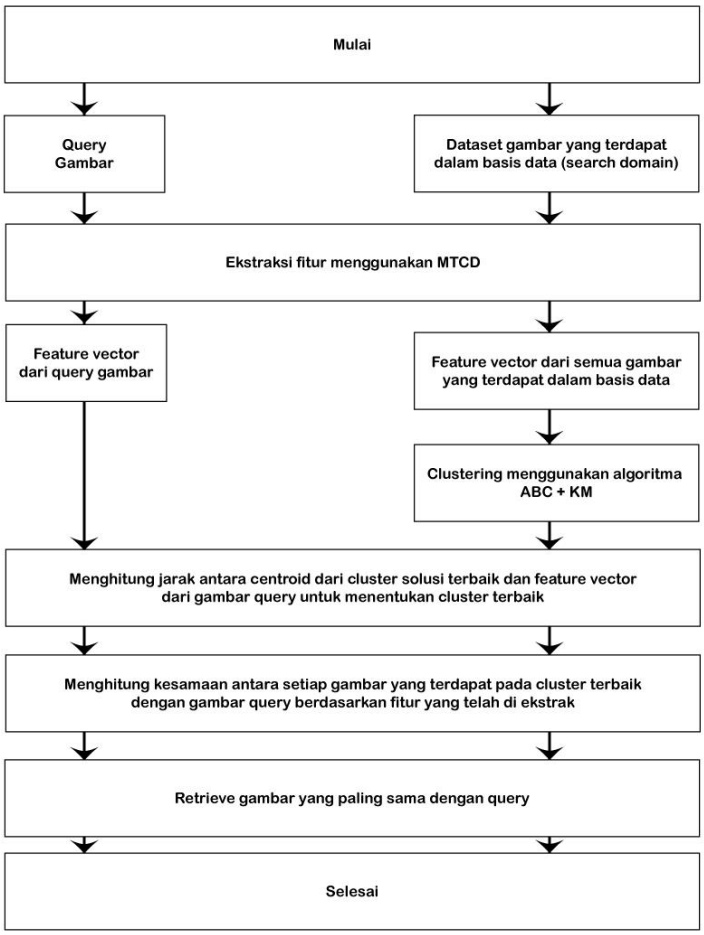
Berdasarkan kekurangan yang dimiliki metode MTCD dan kelebihan yang dimiliki oleh algoritma K-means dan *Artificial Bee Colony,* penulis memiliki hipotesa bahwa jika sistem CBIR dibangun menggunakan algoritma K-means yang dikombinasikan dengan ABC dan descriptor MTCD digunakan untuk ekstaksi fitur, maka dapat menghasilkan sistem CBIR dengan nilai *precision* dan akurasi yang baik. Oleh karena itu dalam penelitian ini penulis mengajukan sistem CBIR yang menggunakan MTCD sebagai *descriptor* dan algoritma K-means yang dikombinasikan dengan *Artificial Bee Colony* sebagai algoritma *clustering*-nya.

2. Metode Penelitian

Pada bagian ini akan dijelaskan bagaimana penelitian dilaksanakan yang meliputi desain dari sistem yang dibuat, penjelasan dari algoritma yang digunakan pada sistem, *dataset* yang digunakan, serta evaluasi performa dari sistem.

2.1 Desain Sistem

Pada bagian ini akan dijelaskan desain dari sistem yang akan diimplementasikan. Penelitian ini melakukan Optimasasi Algoritma K-Means Menggunakan Artificial Bee Colony pada Content-Based Image Retrieval. Untuk ekstraksi fitur gambar digunakan *Multi Texton Co-Occurrence Descriptor* (MTCD).



Gambar 1. Diagram alur desain sistem CBIR yang akan dikembangkan

Berdasarkan Gambar 1 sistem bekerja pertama kali dengan mengekstrak semua fitur gambar yang telah disimpan. Setelah itu *feature vector* hasil ekstraksi akan mulai dikelompokkan menggunakan algoritma ABC+KM. proses pengelompokkan akan terus dilakukan sebanyak jumlah sumber makanan dan iterasi ABC+KM. setelah terbentuk kelompok Langkah selanjutnya adalah menghitung jarak antara centroid pada masing masing kelompok solusi terbaik dengan feature vector dari gambar query untuk menentukan kelompok terbaik. Setelah kelompok terbaik ditentukan, Langkah selanjutnya adalah menghitung kesamaan antara setiap gambar pada kelompok tersebut dengan gambar query. Setelah itu akan ditampilkan gambar – gambar yang sesuai dengan query.

2.2 *Multi Texton Co-Occurrence Descriptor*

Sistem CBIR yang akan dibuat pada penelitian ini menggunakan Multi Texton Co-Occurrence Descriptor (MTCD) sebagai descriptor yang digunakan untuk ekstraksi fitur yang dimiliki oleh gambar. MTCD yang akan digunakan pada sistem CBIR ini merupakan

pengembangan dari MTH (Multi Texton Histogram) hasil penelitian dari Liu dan kawan kawan [10] dengan menambahkan 2 texton baru untuk mendeteksi informasi yang hilang [5].

Metode ekstraksi fitur dalam MTCD terdiri dari 3 langkah, yang pertama adalah mendeteksi fitur local menggunakan MTH, kedua adalah mendeteksi fitur global menggunakan GLCM (Gray Level Co-Occurrence Matrix), dan yang terakhir adalah merepresentasikan semua fitur yang telah terdeteksi menjadi histogram.

2.3 *Gray Level Co-Occurrence Matrix*

Penggunaan GLCM pada MTCD berkontribusi pada penambahan 4 fitur, yakni: *energy, entropy, contrast* dan *correlation*. Langkah pertama dari GLCM adalah mengubah gambar RGB menjadi gambar hitam putih (*grayscale*). Langkah kedua yakni membentuk *co-occurrence matrix*. Langkah ketiga adalah menentukan hubungan spasial antara *reference pixel* dengan neighbor pixel.

2.4 Algoritma *Clustering K-Means*

K-means clustering adalah salah satu algoritma *clustering* yang sering digunakan untuk masalah *clustering* atau pengelompokan data berdasarkan jumlah dan pusat dari cluster yang dibentuk [8]. algoritma ini banyak digunakan karena sederhana dan efisien. *K-means* mengelompokkan objek data menjadi beberapa *cluster* yang telah ditentukan sebelumnya. Masing – masing objek menjadi anggota dari *cluster* tertentu yang dihitung berdasarkan jumlah perhitungan jarak antara setiap objek data dengan *centroid* (pusat cluster) [11].

- Untuk perhitungan jarak *d* antara *node* dengan *centroid*, digunakan perhitungan vector yang disesuaikan dengan jumlah dimensi data seperti pada Persamaan 1.

$$d(z_p, m_k) = \sqrt{\sum_{d=1}^{D} (Z_{pd} - m_{kd})^2} \quad (1)$$

  Di mana *D* adalah dimensi dari vector, Z_{pd} merepresentasikan fitur dalam pola, dan m_{kd} merepresentasikan atribut dari *centroid cluster* ke-*k*.

- Untuk perhitungan perubahan letak *centroid* pada setiap perulangan dapat dilakukan menggunakan Persamaan 2.

$$m_k = \frac{1}{n_k} \sum_{\forall z_p \in C_k} Z_p \quad (2)$$

  Di mana m_k adalah *centroid* dari C_k, C_k adalah *cluster* ke-*K*, n_k adalah jumlah pola data pada cluster, Z_p adalah pola data.

Langkah – Langkah algoritma *K-means* yang digunakan diperjelas dengan pseudocode yang ditunjukkan pada Gambar 2 berikut.

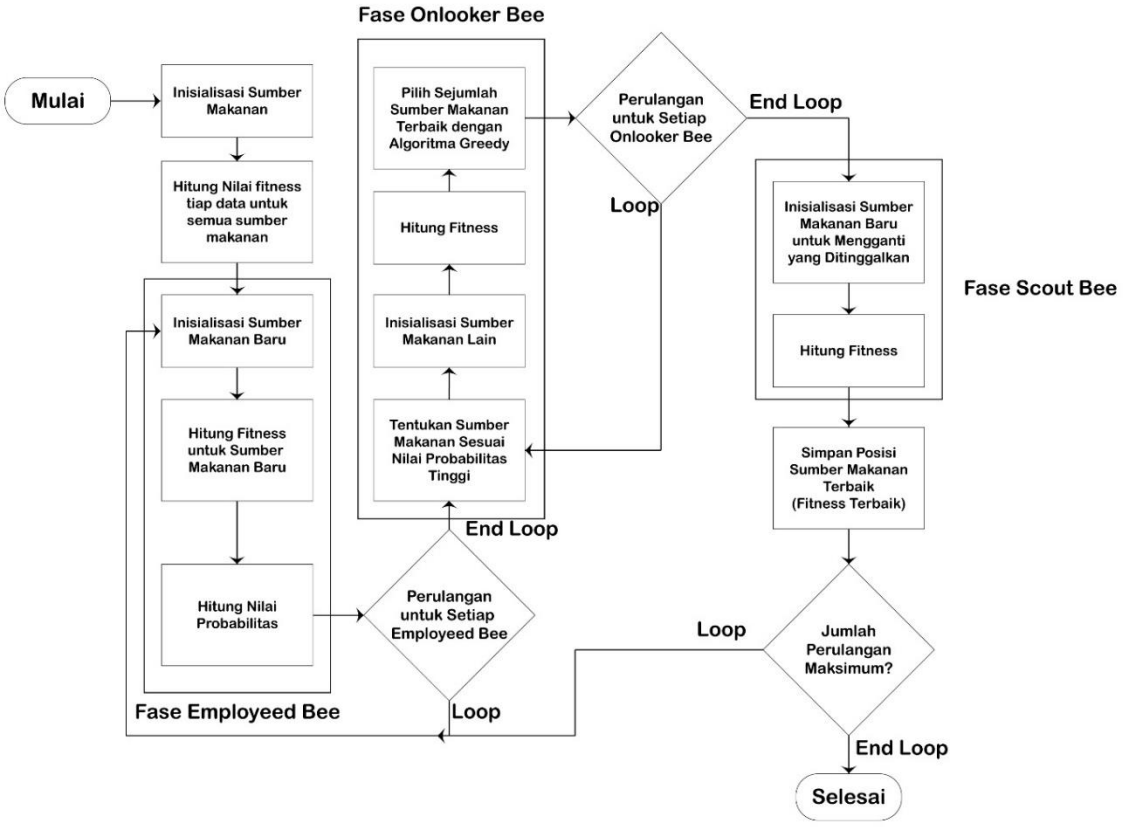
Input **:** K (jumlah dari *cluster* yang akan dibentuk), *set of patterns.*
Output **:** pola yang telah tercluster
1: menentukan jumlah dari *cluster* (K).
2: dari pola data awal, tentukan posisi *cluster* secara acak.
3: hitung *centroid* dari cluster menggunakan persamaan (1).
4: hitung jarak antara data dengan *centroid* menggunakan *Euclidean distance* menggunakan persamaan (2).
5: perbarui informasi *cluster* berdasarkan dengan jarak yang paling kecil.
6: ulangi langkah ke 3, 4, 5, 6 hingga data pada *cluster* tidak terdapat perubahan.
7: outputnya adalah *cluster* yang datanya tidak ada perubahan lokasi *cluster*.

Gambar 2. Pseudocode algoritma K-Means

Memilih *centroid* pada suatu *cluster* merupakan pekerjaan yang rumit karena diatur maupun diacak, apabila nilai inisialisasi yang dilakukan kurang baik maka pengelompokkan data bisa kurang optimal [12].

2.5 Algoritma *Artificial Bee Colony*

Artificial Bee Colony adalah algoritma yang diperkenalkan oleh Dervis Karaboga di tahun 2005, algoritma ini merupakan salah satu dari sekian banyak algoritma optimisasi. *Artificial Bee Colony* merupakan algoritma *meta-heuristic* yang cara kerjanya mengadopsi perilaku koloni lebah dalam pencarian sumber makanan. Sejauh ini ABC telah berhasil dalam mengatasi berbagai masalah dengan berbagai macam pendekatan dan pengaplikasian seperti neural network [13], *clustering* [8], *image processing* [14], *shortest path* [15], dan masih banyak lagi.



Gambar 3. Diagram alur algoritma Artificial Bee Colony

Pada Gambar 3 ini terdapat 3 komponen utama, di antaranya: sumber makanan (*food source*), lebah yang dipekerjakan (*employed bees*), dan lebah yang tak dipekerjakan (*unemployed bees*), lebah yang tak dipekerjakan ini terdiri dari *onlooker* dan *scout bees*. Koloni lebah secara keseluruhan terdiri dari dua bagian, yang pertama merepresentasikan *employed bees* dan yang kedua merepresentasikan *onlooker bees.* Hanya terdapat satu *employed bee* untuk setiap sumber makanan, sehingga jumlah *employeed bees* atau *onlooker bees* sama dengan jumlah sumber makanan yang terdapat di sekitar sarang lebah [16]. Suatu sumber makanan merepresentasikan solusi yang mungkin untuk masalah yang akan dioptimisasikan, jumlah nectar dari sumber makanan mencerminkan kualitas (nilai *fitness*) dari solusi yang bersangkutan.

Secara umum terdapat 5 tahapan proses dari algoritma *Artificial Bee Colony,* yaitu:

1. Fase inisialisasi
2. Fase *Employee bee*
3. Fase *Onlooker bee*
4. Fase *Scout bee*
5. Pemilihan *food source* terbaik

- Fase Inisialisasi

Pada fase inisialisasi algoritma ABC membentuk populasi dari solusi (posisi dari *food source*) menggunakan Persamaan 3. Kemudian solusi tersebut akan mengalami siklus perulangan (1 hingga MCN) dari proses pencarian yang melibatkan *employeed bees, onlooker bees,* dan *scout bees*.

$$x_{ij} = l_i + rand(0,1) * (u_i - u_j) \quad (3)$$

Di mana *i* = 1, . . ., *SN; j* = 1, . . ., *D; SN* merepresentasikan jumlah *food source* dan *D* merepresentasikan jumlah parameter atau dimensi dari masalah, u_i dan i_j adalah batas atas dan batas bawah dari *solution space* dari fungsi tujuan, $rand(0,1)$ adalah angka random distribusi normal dalam rentang [0, 1].

- Fase Employed Bee

Employeed bees mencari solusi lain (*food source* baru) dan membandingkannya dengan solusi yang sebelumnya dengan memeriksa jumlah nectar (nilai *fitness*), kemudian menyimpan solusi yang memiliki nilai *fitness* tertinggi dan menghapus solusi yang memiliki nilai *fitness* lebih rendah. Persamaan 4 digunakan untuk membentuk solusi baru dari solusi sebelumnya. Apabila selesai, *employeed bees* akan membagikan informasi yang paling baru berkenaan dengan kualitas dan posisi *food source* dengan teman mereka.

$$v_{ij} = z_{ij} + \varphi (z_{ij} - z_{kj}) \quad (4)$$

Dimana *i* dan *k* adalah dua sumber makanan yang berbeda, *j* merupakan dimensi, dan φ adalah nilai random yang berdistribusi *uniform* dengan rentang -1 hingga 1.

- Fase Onlooker Bee

Onlooker bees menerima informasi *food source* dari *employeed bees* dan menghasilkan solusi baru dari solusi yang telah ada tergantung pada probabilitas *fitness* dari *food source local*. Seperti halnya *employeed bees,* mereka menyimpan salah satu solusi yang memiliki nilai *fitness* paling tinggi dan menghapus yang memiliki nilai *fitness* lebih rendah. Probabilitas (P_i) dari *food source* dapat dihitung menggunakan Persamaan 5.

$$P_i = 1 - \frac{E_i}{\sum_{j=1}^{SN}(1 - E_j)} \quad (5)$$

Scout bees merubah *food source* saat ini dengan yang baru jika *food source* tersebut diabaikan dalam jumlah yang banyak (melebihi batas yang ditentukan pada parameter).

- Fase Scout Bee

Food source yang diabaikan oleh *bees* akan digantikan dengan *food source* yang baru oleh *scout bees*. Pada algoritma *Artificial Bee Colony*, letak *food source* tidak dapat diperbaiki lebih jauh melalui jumlah siklus yang telah ditentukan sebelumnya sehingga *food source* tersebut diasumsikan telah terlewati atau terabaikan. Jumlah siklus yang ditentukan sebelumnya merupakan *control parameter* yang paling penting pada algoritma ini, yang disebut dengan "*limit*" untuk diabaikan. Misalnya *food source* yang dilupakan adalah z_{ij} dan $j \in \{1, 2, \ldots, D\}$, kemudian *scout bee* menginisialisasi *food source* baru untuk menggantikan z_i. metode untuk mencari *food source* baru sama dengan metode untuk inisialisasi yaitu menggunakan persamaan 3.

Fase ini sama dengan fase sebelumnya, setelah menemukan *food source* baru, akan dihitung nilai *fitnessnya* dengan menggunakan persamaan 8. Setelah itu nilai *fitness food source* baru akan dibandingkan dengan nilai *fitness food source* sebelumnya dengan menggunakan *greedy selection*. Sumber makanan dengan nilai *fitness* terbaik akan disimpan kedalam memori dan akan mengabaikan sumber makanan sebelumnya.

- Pemilihan *Food Source* Terbaik

Fase *Employee bee*, *Onlooker bee* dan *Scout bee* akan terus menerus hingga batas pengulangan atau iterasi dan *food source* terbaik dari setiap *bee* akan dibandingkan hasilnya.

Food source dengan nilai *fitness* paling tinggi akan dipilih sebagai solusi akhir. Penjelasan algoritma secara detail ditunjukkan pada Gambar 4.

1. Inisialisasi populasi dari solusi $x_{i,j}$, $i = 1, \dots, SN,\ \ j = 1, \dots, D$
2. Evaluasi populasi
3. siklus: 1
4. *Repeat*
5. Hasilkan solusi baru $v_{i,j}$ untuk *employeed bees* menggunakan persamaan 4 kemudian evaluasi
6. aplikasikan proses seleksi *greedy*
7. hitung nilai probabilitas $P_{i,j}$ untuk solusi $x_{i,j}$ dengan menggunakan persamaan 5
8. hasilkan solusi baru $v_{i,j}$ untuk *onlooker bees* dari solusi $x_{i,j}$ yang dipilih berdasarkan perhitungan $P_{i,j}$ kemudian evaluasi
9. aplikasikan proses seleksi *greedy*
10. tentukan solusi yang diabaikan untuk *scout bees* , jika ada, gantikan dengan solusi baru yang dihasilkan secara acak $x_{i,j}$
11. Simpan solusi terbaik
12. siklus = siklus + 1
13. ulangi hingga siklus = MCN

Gambar 4. Pseudocode algoritma Artificial Bee Colony

3. Hasil Penelitian dan Pembahasan

3.1 Dataset

Dataset yang digunakan pada penelitian ini menggunakan 2 jenis dataset pada Gambar 5 dan Gambar 6, yaitu: corel 10.000 database dan Batik Dataset. Penggunaan 2 jenis dataset ini dilakukan untuk mengetahui performa sistem dalam menangani dataset dalam jumlah besar dan kecil.

Corel 10.000 database terdiri dari 100 kategori dan masing – masing kategori terdiri dari 100 gambar dengan ukuran 192x128 atau 128x192 dengan format JPEG. Batik dataset terdiri dari 50 kelas atau kategori masing - masing kategori terdiri dari 6 data, sehingga total data dari Batik dataset adalah 300 data.

Gambar 5. Contoh data pada dataset Batik

Gambar 6. Contoh data pada dataset Corel-10.000

3.2. Evaluasi Performa

Pada ranah *information retrieval*, *precision* dan *recall* sering digunakan untuk menghitung performa sistem yang dibangun. *Precision* (biasa juga disebut *positive predictive value*) digunakan untuk mengetahui apakah gambar yang di *retrieve* benar. *Recall* digunakan untuk mengevaluasi kemampuan dari sistem dalam menemukan gambar yang relevel pada database. Perhitungan *precision* dan *recall* ditunjukan pada Persamaan 6 dan Persamaan 7 berikut.

$$precision = \frac{A}{A+B} \tag{6}$$

$$Recall = \frac{A}{A+C} \tag{7}$$

Di mana,
A = jumlah gambar yang relevan ter *retrieve*
B = jumlah gambar yang tidak relevan
C = jumlah gambar yang relevan yang tidak ter *retrieve*

3.3. Skenario Uji Coba

Testing dilakukan pertama kali pada dataset corel 10.000. dataset ini memiliki 10.000 data yang terdiri dari 100 kelas, di mana masing masing kelas memiliki 100 data. Dari 100 data ini, 50 data di ambil secara acak dari masing masing kelas untuk dijadikan data test, sehingga kami memiliki 5.000 data test untuk dataset corel 10.000.

Dataset kedua yang digunakan untuk testing merupakan dataset batik dengan jumlah data sebanyak 300 yang terdiri dari 50 kelas, masing masing kelas terdiri dari 6 data. Dari 6 data ini, 1 data diambil secara acak dari masing masing kelas untuk dijadikan data test, sehingga kami memiliki 50 data test untuk dataset batik. Kami memberlakukan beberapa test scenario untuk mengetahui:

1. Berapa nilai SN (sumber makanan) yang optimum untuk menghasilkan nilai precission dan recall terbaik.
2. Berapa nilai iterasi K-Means yang optimum untuk menghasilkan nilai precission dan recall terbaik.
3. Berapa nilai cluster K-means yang optimum untuk menghasilkan nilai fitness terbaik.

Pada tahap awal uji coba dilakukan untuk menetapkan variable utama pada program, yaitu jumlah-*cluster*, besaran dimensi-yang-digunakan, *food source* (SN), serta jumlah perulangan algoritma *Artificial Bee Colononi*-nya. Banyaknya *cluster* yang digunakan sesuai-dengan hasil dari percobaan *image retrieval* sebelumnya dengan hanya menggunakan *K-means* tanpa ditambah dengan algoritma ABC. Ditunjukkan pada Tabel 1 untuk dataset Corel 10.000 dan Tabel

2 untuk dataset Batik. Sedangkan jumlah sumber makanan dan iterasi ABC akan divariasikan nilai variabelnya antara 20 – 100 untuk uji coba selanjutnya.

Tabel 1. Nilai Precission dan Recall berdasarkan jumlah cluster pada dataset Corel-10.000

	Precission	Recall
MTCD	0.69	0.08
MTCD + K-Means		
3 cluster	0.63	0.07
5 cluster	0.69	0.08
7 cluster	0.72	0.08
9 cluster	0.72	0.08
11 cluster	0.73	0.08
13 cluster	0.69	0.08
15 cluster	0.67	0.07

Terlihat pada Tabel 1 nilai cluster terbaik untuk dataset corel 10.000 yaitu 11. Nilai cluster di atas 11 sudah mengalami penerununan performa pada *precission* dan *recall* – nya. Sehingga nilai K = 11 ini digunakan sebagai nilai K pada sistem ABC+KM.

Tabel 2. Nilai precission dan recall berdasarkan jumlah cluster pada dataset Batik

	Precission	Recall
MTCD	0.514	0.944
MTCD + K-Means		
3 cluster	0.363	0.666
5 cluster	0.514	0.944
7 cluster	0.545	1
9 cluster	0.393	0.722
11 cluster	0.302	0.544
13 cluster	0.302	0.544
15 cluster	0.363	0.666

Pada uji coba di Tabel 2 terlihat bahwa nilai *precission* dan *recall* tertinggi terdapat pada nilai K sebesar 7. Nilai K = 7 digunakan pada implementasi program selanjutnya dengan menggunakan dataset Batik.

3.4 Uji Coba Kinerja Sistem

Uji kinerja dilakukan untuk melakukan pengujian kinerja program dalam masalah pembagian-data menjadi beberapa *cluster*. Pengujian kinerja program ini dilakukan dengan mencari angka *fitness* pada *food source* yang telah terbentuk setelah diaplikasikan algoritma ABC+KM, beberapa variasi digunakan sebagai parameter jumlah total *food source* serta iterasi dari algoritma *Artificial Bee Colony*. Selain nilai fitness uji kinerja juga dilakukan dengan mencari nilai *precission* dan *recall* untuk hasil *retrieval* untuk mengetahui seberapa presisi performa program dengan menggunakan algoritma ABC+KM.

Nilai *fitness* pada suatu *food source* dihitung menggunakan *Sum of Squared Error* (SSE), *fitness* itu sendiri adalah nilai-total-dari-jarak-sebuah-*node*-*data*-menujut-*cluster*-terdekat dari data itu. Apabila nilai *fitness*-nya semakin kecil maka kualitas *cluster*-nya semakin baik.

Tabel 3 menyajikan-perbandingan-angka *fitness* menggunakan beberapa-variasi variable dari-implementasi-algoritma-ABC+KM pada dataset Corel-10.000 dan pada Tabel 4 menunjukkan nilai *fitness* pada dataset Batik. Nilai *fitness* diperoleh dari rata – rata nilai yang dihasilkan dalam percobaan dengan masing – masing besaran iterasi dan sumber makanan (SN).

Tabel 3. Nilai Fitness Uji ABC+KM dataset Corel-10.000

Fitness	Sumber Makanan						
	KM	20	30	50	75	100	AVG
KM	5104549.468						

Iterasi	25	5066584.705	5069251.960	5066582.315	5068417.237	5066714.746	5067510.193
	50	5073633.627	5066015.856	5055312.047	5067827.459	5066707.955	5065899.389
	75	5075544.369	5066516.096	5056116.920	5065674.882	5053588.417	5063488.137
	100	5053613.285	5066549.762	5055939.560	5053609.919	5053588.417	5056660.189
	AVG	5067343.997	5067083.419	5058487.711	5063882.374	5060149.884	

Setelah melakukan percobaan dengan menggunakan beberapa parameter yang telah ditentukan pada Tabel 3 di atas, nilai *fitness* secara keseluruhan dengan menggunakan algoritma *K-means* saja, yaitu **5104549.468**, nilai tersebut lebih besar dari pada menggunakan algoritma ABC+KM, nilai paling kecil dihasilkan oleh pengujian dengan menggunakan SN sebanyak 100 dan 100 jumlah iterasi (**5053588.417**) nilai paling besar dihasilkan oleh pengujian dengan menggunakan SN sebanyak 20 serta 75 iterasi (**5075544.369**), hasil ini memperlihatkan bahwa kualitas *cluster* yang terbentuk lebih baik dengan menggunakan algoritma ABC+KM.

Tabel 4. Nilai Fitness Uji ABC+KM dataset batik

Fitness		Sumber Makanan						
		KM	20	30	50	75	100	AVG
	KM	210280.798						
Iterasi	25		195698.926	191876.573	193693.046	191176.216	191582.710	192805.4942
	50		194280.961	193086.959	192291.956	191292.535	191121.267	192414.7356
	75		196209.513	195405.723	193114.183	191582.710	190581.407	193378.7072
	100		193063.453	192089.450	191111.296	191206.222	191051.708	191704.4258
	AVG		194813.2133	193114.6763	192552.6203	191314.4208	191084.273	

Setelah melakukan percobaan dengan menggunakan beberapa parameter yang telah ditentukan pada dataset Batik, nilai *fitness* secara keseluruhan dengan menggunakan algoritma *K-means* saja, yaitu **210280.798**, nilai tersebut lebih besar dari pada menggunakan algoritma ABC+KM, nilai paling kecil dihasilkan oleh pengujian dengan menggunakan SN sebanyak 100 dan 75 jumlah iterasi (**190581.407**) nilai paling besar dihasilkan oleh pengujian dengan menggunakan SN sebanyak 20 serta 75 iterasi (**196209.513**), hasil ini memperlihatkan bahwa kualitas *cluster* yang terbentuk lebih baik dengan menggunakan algoritma ABC+KM.

Setelah itu mengkalkulasikan nilai *precission* dan *recall* dilakukan-untuk-mengetahui seberapa presisi *retrieval* yang dihasilkan oleh sistem. Untuk menghitung nilai *precission* dan *recall* menggunakan Persamaan 7 dan Persamaan 8 yang telah dijelaskan pada bab sebelumnya.

Tabel 5 dan Tabel 6 masing – masing menunjukkan nilai *precission* dan *recall* untuk dataset Corel-10.000. persamaan 7 dan 8 menujukkan metode perhitungan kedua nilai tersebut. Nilai *precission* hasil dari algoritma ABC+KM ternyata lebih unggul sebesar 0.27 poin pada angka **0.757** pada sumber makanan (SN) 50 dan iterasi *K-means* 25 dibandingkan dengan hanya *K-means* yang berada pada nilai **0.73**.

Tabel 5. Nilai Precission Uji ABC + K-Means dataset Corel-10.000

Precission		Sumber Makanan						
		KM	20	30	50	75	100	AVG
	KM	0.73						
Iterasi	25		0.605	0.727	0.757	0.605	0.727	0.684

	50	0.605	0.605	0.605	0.605	0.605	0.605
	75	0.605	0.605	0.605	0.605	0.605	0.605
	100	0.727	0.727	0.605	0.605	0.605	0.653
	AVG	0.635	0.666	0.643	0.605	0.635	

Tabel 6. Nilai Recall Uji ABC + K-Means dataset Corel-10.000

Recall		Sumber Makanan						
		KM	20	30	50	75	100	AVG
	KM	0.08						
Iterasi	25		0.06	0.08	0.08	0.06	0.08	0.07
	50		0.06	0.06	0.06	0.06	0.06	0.06
	75		0.06	0.06	0.06	0.06	0.06	0.06
	100		0.08	0.08	0.06	0.06	0.06	0.07
	AVG		0.07	0.07	0.07	0.06	0.07	

Nilai *recall* terbaik untuk penerapan algoritma ABC+KM muncul pada sumber makanan 30 dan iterasi *K-means* 25 dengan nilai **0.08**. nilai tersebut sama besar dengan nilai *recall* yang hanya menggunakan *K-means* yaitu sebesar **0.08**.

Selanjutnya Tabel 7 dan Tabel 8 masing – masing menunjukkan nilai *precision* dan *recall* untuk dataset Batik. Berbeda dengan dataset Corel-10.000, pada dataset Batik nilai *precission* dengan menggunakan ABC+KM masih kalah dengan yang hanya menggunakan *K-means*, nilai *precission* penggunaan ABC+KM berada di angka **0.514** dengan sumber makanan 50 dan iterasi 75. Sedangkan nilai dengan hanya menggunakan *K-means* adalah **0.545**.

Tabel 7. Nilai Precission Uji ABC + K-Means dataset Batik

Precission		Sumber Makanan						
		KM	20	30	50	75	100	AVG
	KM	0.545						
Iterasi	25		0.423	0.423	0.363	0.423	0.423	0.411
	50		0.363	0.423	0.423	0.423	0.393	0.405
	75		0.423	0.423	0.514	0.393	0.454	0.441
	100		0.393	0.423	0.423	0.423	0.423	0.417
	AVG		0.400	0.423	0.430	0.415	0.423	

Tabel 8. Nilai Recall Uji ABC + K-Means dataset Batik

Recall		Sumber Makanan						
		KM	20	30	50	75	100	AVG
	KM	1						
Iterasi	25		0.777	0.777	0.655	0.777	0.777	0.752
	50		0.666	0.777	0.777	0.777	0.722	0.743
	75		0.777	0.777	0.933	0.722	0.833	0.808
	100		0.722	0.777	0.777	0.777	0.777	0.766
	AVG		0.735	0.777	0.785	0.763	0.777	

Untuk nilai *recall* penggunaan algoritma ABC+KM juga kalah dengan hanya menggunakan *K-means*. Nilai *recall* penggunaan ABC+KM berada pada nilai **0.933** dengan sumber makanan 50 dan iterasi 75. Sedangkan nilai hanya menggunakan *K-means* adalah **1**.

Tabel 9. Perbandingan nilai performa untuk masing - masing metode

Data	Performace	Method		
		MTCD	MTCD+KM	MTCD+ABC+KM (proposed method)
Batik	precission	0.514	0.545	0.545
	recall	0.944	1	0.933
	fitness	-	210280.798	190581.407
Corel 10K	precission	0.69	0.73	0.757

recall	0.08	0.08	0.08
fitness	-	5104549.468	5053588.417

Tabel 9 merupakan tabel perbandingan nilai performa untuk masing – masing dataset. Dalam penelitian ini terdapat 3 nilai performa yang diukur, yaitu *precission, recall*, dan *fitness.*

Untuk dataset batik terlihat tidak ada kenaikan nilai *precission* dalam penggunaan algoritma ABC+KM. justru untuk nilai *recall* terdapat penurunan performa. Namun pada nilai fitness memiliki performa yang cukup bagus, hal ini dapat dikatan dengan menggunakan ABC+KM cluster yang terbentuk lebih baik. Tetapi cluster yang baik tidak menjamin untuk memperbaiki nilai *precission* dan *recall.*

Untuk dataset Corel-10.000 terdapat kenaikan pada nilai *precission* meskipun kenaikannya tidak cukup banyak hanya sebesar 0.02. untuk nilai *recall*, penggunaan algoritma ABC+KM tidak menimbulkan efek apapun, nilai *recall* sebelum dan sesudah penggunaan ABC+KM sama. Nilai *fitness* pada dataset ini memiliki nilai yang cukup bagus setelah penggunaan algoritma ABC+KM, nilainya menurun dari **5104549.468** menjadi **5053588.417**, hal ini menunjukkan cluster yang terbentuk lebih baik dengan penggunaan ABC+KM dari pada hanya menggunakan *K-means*. Untuk kasus pada dataset ini nilai *fitness* yang baik berpengaruh pada nilai *precission* yang meningkat.

4. Kesimpulan

Penelitian ini mengajukan metode baru yaitu pengoptimisasian algoritma clustering k-means menggunakan Artificial Bee Colony pada sistem CBIR. Algoritma k – means clustering memiliki beberapa kekurangan, salah satunya adalah tidak dapat memberikan hasil clustering yang optimal karena cluster dari k-means bergantung pada centroid awal cluster [11]. Metode yang diajukan pada penelitian ini bertujuan untuk menutupi kekurangan k-means dengan melakukan global search atau clustering (penentuan food source) terlebih dahulu menggunakan Artificial Bee Colony. Pada penelitian ini algoritma k-means berfungsi untuk mengkategorikan gambar kedalam suatu grup, sementara itu algoritma ABC melakukan global search pada solution space. Sebagai dampaknya metode yang diajukan membantu dalam mempersempit search space serta mempersingkat waktu komputasi untuk membandingkan semua gambar pada database dengan image query.

Berdasarkan pengujian dan evaluasi yang telah dilakukan, diperoleh kesimpulan sebagai berikut:

1. Algoritma ABC+KM dapat membentuk *cluster* yang baik, ditunjukkan dengan nilai *fitness* yang dihasilkan. nilai terendah = **5075544.369**, tertinggi = **5053588.417** untuk dataset Corel-10.000, dan untuk dataset Batik nilai terendah = **196209.513**, tertinggi = **190581.407.** semua nilai itu lebih baik dari nilai *fitness* yang hanya menggunakan *K-means*.
2. Nilai *precission* yang dihasilkan oleh algoritma ABC+KM memiliki nilai lebih baik (**0.757**) dibandingkan dengan hanya menggunakan *K-means* (**0.73**) untuk dataset Corel-10.000. sedangkan untuk dataset Batik penggunaan ABC+KM menghasilkan nilai *precission* yang tidak lebih baik (**0.514**) dibandingkan dengan hanya menggunakan *K-means* (**0.545**).
3. Nilai recall yang dihasilkan oleh algotima ABC+KM memiliki nilai yang sama besar dengan hanya menggunakan *K-means* yaitu **0.08** untuk dataset Corel-10.000. sedangkan untuk dataset Batik penggunaan ABC+KM memiliki nilai yang tidak lebih baik (**0.933**) dibandingkan dengan hanya menggunakan *K-means* (**1.0**).

Untuk memperoleh hasil yang bisa jadi lebih baik untuk penelitian selanjutnya terdapat beberapa saran melalui penelitian ini, yaitu:

1. Penambahan iterasi untuk algoritma ABC dan variasi jumlah sumber makanan, disamping menambah jumlah uji coba keseluruhan yang dilakukan.
2. Menggunakan metode lain untuk perhitungan nilai *fitness* selain menggunakan *Sum of Squared Error* (SSE).
3. Melakukan *preprocessing* data yang lebih baik agar hasil yang diperoleh lebih akurat.

Daftar Notasi

SN	Jumlah sumber makanan
D	Jumlah dimensi data
K	Jumlah *Cluster*

m_k	*Centroid*
C_k	*Cluster* ke-*K*
n_k	Jumlah pola data pada *cluster*
Z_p	Pola data
i	Merepresentasikan banyak *SN* {1, 2, 3, ..., *SN*}
j	Merepresentasikan banyak dimensi {1, 2, 3, . . ., *D*}
u_i	Batas atas *solution space*
u_j	Batas bawah *solution space*
P_i	Nilai probabilitas
n_{ij}	*Food Source* baru
$\|Z_i - C_j\|^2$	*Sum of Squared Error* (SSE) data Z_i ke *centroid* C_j
v_{ij}	Kandidat posisi sumber makanan baru berdasarkan yang lama
φ	Nilai acak dengan rentang [-1, 1]
P	Nilai probabilitas
E	Nilai *fitness*

Referensi

[1] Y. Liu, D. Zhang, G. Lu, and W. Ma, "A survey of content-based image retrieval with high-level semantics," vol. 40, pp. 262–282, 2007.

[2] T. Quack, M. Ullrich, L. Thiele, and B. S. Manjunath, "Cortina : A System for Large-scale , Content-based Web Image Retrieval," 2004.

[3] J. R. Smith and S. Chang, "VisualSEEk : a fully automated content-based image query system," 1996.

[4] J. Ze, "SIMPLIcity : Semantics-sensitive Integrated Matching for Picture LIbraries 1 Introduction," pp. 1–25.

[5] U. M. Malang, "Image Retrieval Using Multi Texton CO-," vol. 67, no. 1, pp. 103–110, 2014.

[6] J. Rejito, A. S. Abdullah, and D. Setiana, "Image indexing using color histogram and k-means clustering for optimization CBIR in image database Image Indexing using Color Histogram and k -means Clustering for Optimization CBIR in Image Database," 2017.

[7] "No Title," 2005.

[8] D. Karaboga and C. Ozturk, "A novel clustering approach : Artificial Bee Colony (ABC) algorithm," vol. 11, pp. 652–657, 2011.

[9] A. A. Amory, R. Sammouda, H. Mathkour, and R. M. Jomaa, "A Content Based Image Retrieval Using K-means Algorithm," pp. 221–225, 2012.

[10] G. Liu, L. Zhang, Y. Hou, Z. Li, and J. Yang, "Image retrieval based on multi-texton histogram," *Pattern Recognit.*, vol. 43, no. 7, pp. 2380–2389, 2010.

[11] A. Alharan, A. Al-haboobi, and H. T. Kurmasha, "Content-Based Image Retrieval Hybrid Approach using Artificial Bee Colony International Journal of Sciences : Content-Based Image Retrieval Hybrid Approach using Artificial Bee Colony and K-means Algorithms," no. June, 2016.

[12] G. Armano and M. R. Farmani, "Clustering Analysis with Combination of Artificial Bee Colony Algorithm and k -Means Technique," vol. 6, no. 2, pp. 141–145, 2014.

[13] A. Bee, C. Abc, and T. Artificial, "Neural Networks."

[14] C. Chidambaram and H. S. Lopes, "A New Approach for Template Matching in Digital Images Using an Artificial Bee Colony Algorithm."

[15] P. Mansouri and B. Asady, "Solve Shortest Paths Problem by Using Artificial Bee Colony Algorithm Solve Shortest Paths Problem by Using Artificial Bee Colony Algorithm," no. January 2014, 2016.

[16] D. Karaboga and B. Basturk, "Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization," no. December, 2015.