

BAB II

METODE PENELITIAN

2.1 Objek Penelitian

Penyakit pada tanaman biasanya disebabkan oleh jamur dan bakteri yang dapat berdampak negatif terhadap produksi dan kualitas tanaman. Penyakit tanaman padi dapat dikenali dari perubahan fisik pada permukaan daun (Sitompul, Okprana and Prasetyo, 2022). Dalam proses budidaya, banyak tanaman padi yang rentan terhadap serangan penyakit serta hama. Pada kebanyakan kasus, ketika tanaman padi diserang oleh hama dan penyakit petani akan langsung mengaplikasikan pestisida atau memberikan penanganan yang sering kali tidak tepat dengan penyakit dan hama yang menyerang tanaman padi. (Jinan and Hayadi, 2022).

Pada penelitian ini berfokus pada Jaringan Syaraf Tiruan terkait klasifikasi penyakit pada tanaman padi dengan menggunakan metode *Backpropagation* dan optimasi *Bee Colony Optimization*. Sistem ini akan mengolah data dan informasi tentang gejala dan penyakit pada tanaman padi. Adapun jenis-jenis penyakit tanaman padi beserta gejala yang digunakan dapat dilihat pada Tabel 2.1.

Tabel 2.1 Penyakit dan Gejala

No	Penyakit	Gejala
1.	Tungro	<ul style="list-style-type: none">- Daun kuning kemerah – merah- Daun muda menjadi belang/garus hijau pucat- Bulir bercak cokelat dan beratnya kurang dibanding normal- Kerdil- Jumlah anakan sedikit berkurang- Pertumbuhan akar tidak sempurna
2.	Blas	<ul style="list-style-type: none">- Bercak berbentuk belah ketupat- Bercak cokelat kehitaman pada batang- Batang mudah patah- Malai hampa atau tidak berisi
3.	Kresek	<ul style="list-style-type: none">- Daun berwarna putih kekuningan- Layu- Batang berwarna cokelat
4.	Bercak daun cokelat	<ul style="list-style-type: none">- Daun garis memanjang atau oval- Bercak muda berbentuk bulat kecil- Bercak berwarna cokelat gelap- Bercak tua berukuran lebih besar (0,4 – 1 cm x -,1 – 02 cm)- Bercak berwarna kuning di sekelilingnya- Bulir berwarna cokelat kehitaman
5.	Bercak sempit cokelat	<ul style="list-style-type: none">- Pada daun dan pelepah daun terdapat bercak cokelat yang sempit- Varietas yang tahan bercak berukuran 0,2 – 1 cm x 0,1 cm, berwarna cokelat gelap- Varietas bercak lebih besar & berwarna cokelat terang

No	Penyakit	Gejala
6.	Bercak Garis	<ul style="list-style-type: none"> - Muncul garis yang kebasah – basahan diantara urat daun - Garis memanjang dan menjadi cokelat dengan lingkaran kuning di sekelilingnya - Berlendir - Lendir yang kering berbentuk butiran kecil pada garis luka
7.	Hangus Palsu	<ul style="list-style-type: none"> - Bulir padi menjadi gumpalan spora yang ukurannya sampai 1 cm - Gumpalan spora menjadi hijau gelap - Daun yang menguning menjadi kering
8.	Kerdil Hampa	<ul style="list-style-type: none"> - Daun jadi kasar, tidak teratur - Bulir padi hanya sedikit yang berisi - Daun menguning dan terpilin - Tanaman menjadi kerdil
9.	Kerdil Rumput	<ul style="list-style-type: none"> - Tanaman padi sangat kerdil - Daun berwarna kuning / tetap hijau - Tetap berbunga, tapi bulir padi tidak berisi
10.	Busuk Batang	<ul style="list-style-type: none"> - Pelepah daun terlihat bercak basah berbentuk bulat - Bercak pada bagian tengah berwarna abu – abu & bagian tepi berwarna cokelat - Pembusukan batangnya dari pangkal hingga atas
11.	Kerdil Kuning	<ul style="list-style-type: none"> - Warna daun dari kuning kehijauan ke kuning keputihan - Bulir padi hampa - Kerdil / pendek

2.2 Data Penelitian

Data yang digunakan pada penelitian ini ialah gejala penyakit pada tanaman padi di lingkungan Samarinda. Data tersebut berjumlah 1094 yang didapatkan dengan memberikan kuesioner berisi gejala pada penyakit tertentu kepada petani yang disesuaikan dengan gejala penyakit sesuai literatur. Terdapat 11 penyakit yang menyerang tanaman padi beserta gejalanya seperti yang terlampir pada Tabel 2.1. Teknik pengumpulan data agar mendapat informasi yang akurat pada penelitian ini ialah dengan melakukan wawancara, observasi, dan dokumentasi.

Untuk memperoleh informasi yang akurat dan relevan, penelitian ini juga melibatkan wawancara dengan petani di Lempake, Kec. Samarinda Utara, Kota Samarinda, Bukit Raya, Kec. Tenggarong Seberang, Kabupaten Kutai Kartanegara. Observasi dilakukan untuk menilai kondisi fisik dan lingkungan di sekitar setiap lokasi yang dipertimbangkan. Keseluruhan proses pengumpulan data ini membentuk dasar yang kokoh untuk klasifikasi jenis penyakit pada tanaman padi menggunakan algoritme *Backpropagation* dan optimasi *Bee Colony Optimization*.

2.3 Jaringan Syaraf Tiruan

Jaringan Syaraf Tiruan adalah permodelan data yang sangat efektif yang mampu merepresentasikan dan menangkap hubungan yang kompleks antara *input* dan *output*. Hal ini disebabkan oleh kemampuannya untuk menyelesaikan berbagai masalah dengan relatif mudah, ketahanannya terhadap variasi data *input*, dan kecepatan dalam mengeksekusi dan menginisialisasi sistem yang kompleks (Satria, 2020).

Kemampuan Jaringan Syaraf Tiruan (JST) sangat efektif, seperti yang terlihat dari beberapa aplikasinya. Jaringan Syaraf Tiruan sangat cocok untuk digunakan dalam Klasifikasi, Optimasi, Asosiasi, dan *Self Organizing* (Fitrianiingsih Hasan, Kusri and Al Fatta, 2019). Konsep Jaringan Syaraf Tiruan melibatkan pemrosesan sinyal *input* untuk menghasilkan *output* melalui neuron yang tersusun dalam tiga lapisan: lapisan *input*, lapisan tersembunyi, dan lapisan *output* (Hasanati *et al.*, 2020).

2.4 Klasifikasi

Klasifikasi adalah suatu pembelajaran terawasi yang dapat digunakan untuk menganalisis data dan membangun model yang memisahkan data ke dalam kelas-kelas berdasarkan kriteria tertentu (Cervantes *et al.*, 2020). Klasifikasi sebagian besar digunakan untuk memprediksi kelas pada sebuah label, dengan mengklasifikasikan data (membangun model) menurut sekumpulan nilai pelatihan (label kelas) untuk mengklasifikasikan atribut tertentu (Tangkelayuk & Mailoa, 2022).

2.5 Multiclass

Klasifikasi multi-kelas adalah metode klasifikasi untuk mendeteksi satu atau lebih kelas dari sebuah sampel dan setiap sampel hanya dapat dilabeli dengan satu kelas (Ningsih *et al.*, 2024).

2.5.1 One vs One

One-Versus-One merupakan sebuah metode klasifikasi yang bekerja dengan cara membandingkan dua kelas yang berbeda satu sama lain (Kartika Delimayanti *et al.*, 2021).

Contoh :

Terdapat kelas X, Y, dan Z

- Model 1: Melatih untuk membedakan Kelas X versus Kelas Y.
- Model 2: Melatih untuk membedakan Kelas X versus Kelas Z.
- Model 3: Melatih untuk membedakan Kelas Y versus Kelas Z.

2.5.2 One vs All

One-Versus-All adalah sebuah metode klasifikasi yang membandingkan satu kelas dengan semua kelas lain secara bersamaan (Kartika Delimayanti *et al.*, 2021).

Contoh :

Terdapat kelas X, Y, dan Z

- Model 1: Melatih untuk membedakan Kelas X (positif) versus Kelas Y dan Z (negatif).
- Model 2: Melatih untuk membedakan Kelas Y (positif) versus Kelas X dan Z (negatif).
- Model 3: Melatih untuk membedakan Kelas Z (positif) versus Kelas X dan Y (negatif).

2.6 K-Fold

Pengujian *Cross Validation K-Fold* digunakan untuk mengevaluasi kinerja metode algoritme (Adiba, Akram Nur Risal & Tahir, 2023). *K-Fold* digunakan untuk membagi data menjadi data uji

dan data latih dengan tujuan mengurangi kemungkinan terjadinya pembagian data yang menguntungkan, seperti yang bisa terjadi saat menggunakan metode split data sederhana (Pradema Sanjaya, Pribadi & Prastya, 2022). Konsep *K-Fold* dapat dilihat pada Gambar 2.1.

Iterasi 1	Test	Train	Train	Train	Train
Iterasi 2	Train	Test	Train	Train	Train
Iterasi 3	Train	Train	Test	Train	Train
Iterasi 4	Train	Train	Train	Test	Train
Iterasi 5	Train	Train	Train	Train	Test

Gambar 2.1 *K-Fold*

2.7 Backpropagation

Backpropagation merupakan algoritme pembelajaran terawasi yang biasa digunakan oleh *perceptron* berlapis untuk mengoptimalkan bobot dengan menghubungkan neuron-neuron di lapisan tersembunyi. Algoritme ini menggunakan kesalahan *output* untuk mengubah nilai bobot-bobot tersebut dalam arah mundur (backward) (Hutabarat et al., 2021).

Untuk mendapatkan kesalahan *output* ini, langkah perambatan maju (*forward propagation*) harus dilakukan terlebih dahulu. Selama perambatan maju, neuron-neuron diaktifkan dengan menggunakan fungsi aktivasi yang bisa didiferensiasikan, seperti sigmoid (Jayanti, Lumbanbatu & Ramadani, 2021).

Backpropagation terdiri dari 3 fase, yaitu *feedforward*, *backward* dan penyesuaian bobot. Berikut rumus – rumus yang digunakan pada fase tersebut (Rolimarch Pratama & Darmawan, 2021). Fase *feedforward* setiap unit *input* menerima sinyal dan mengirimkannya ke unit tersembunyi, yang kemudian menghitung bobot sinyal *input* tersebut (z_j) digunakan rumus pada Persamaan 2.1.

$$z_{in_j} = v_{0j} + \sum_i x_i v_{ij} \quad (2.1)$$

Keterangan :

z_{in_j} = Nilai unit tersembunyi ke-j

x_i = Unit *input* ke-i untuk sebuah unit *input*, sinyal *input* dan sinyal *output* adalah sama yaitu x

v_{0j} = Bobot pada unit di layer ke tersembunyi ke-j

v_{ij} = Bobot pada unit ke-i di layer tersembunyi ke-j

Menghitung *output* pada *hidden layer* menggunakan fungsi aktivasi digunakan rumus pada Persamaan 2.2.

$$z_{out} = \frac{1}{1 + \exp(z_{in_j})} \quad (2.2)$$

Keterangan :

z_{out} = *Output* akhir pada *hidden layer*

z_{in_j} = *Ouput* dari *input* untuk unit ke-j

Menghitung bobot sinyal *input* di unit *output* Y_k digunakan rumus pada Persamaan 2.3 dan 2.4.

$$y_{in_k} = w_{0k} + \sum_i z_j w \quad (2.3)$$

$$y_k = f(y_{in_k}) \quad (2.4)$$

Keterangan :

Y_{in_k} = Nilai *output* tersembunyi ke-k

Y_k = Nilai unit *output* ke-k menggunakan fungsi aktivasi

W_{0k} = Bobot unit tersembunyi ke unit *ouput* ke-k

W_{jk} = Bobot unit tersembunyi ke-j ke unit *output* ke-k

Fase *Backward* menghitung faktor δ unit *output* berdasarkan kesalahan di setiap unit *output* digunakan rumus pada Persamaan 2.5.

$$\delta_k = (t_k - y_k) f^1(y_{in_k}) \quad (2.5)$$

Keterangan :

δ_k = Nilai *error* unit *output*

t_k = Nilai target *output*

Menghitung perbaikan pada bobot w_{jk} yang menghubungkan *hidden layer* dan *input layer* digunakan rumus pada Persamaan 2.6.

$$\Delta w_{jk} = \alpha \delta_k z_j \quad (2.6)$$

Keterangan :

δ_k = Nilai *error* unit *output*

α = *Learning Rate*

Δw_{jk} = Perubahan bobot unit tersembunyi ke-i ke unit *output* ke-k

z_j = Luaran di *hidden layer*

Menghitung faktor δ unit tersembunyi berdasarkan kesalahan di setiap unit tersembunyi z_j digunakan rumus pada Persamaan 2.7.

$$\delta_{in_j} = \delta_k \times w_{jk} \quad (2.7)$$

Keterangan :

δ_{in_j} = Nilai *error* pada *hidden layer*

δ_k = Nilai *error* unit *output*

w_{jk} = Bobot unit tersembunyi ke-i ke unit *output* ke-k

Menghitung hasil bobot δ_j di *hidden layer* digunakan rumus pada Persamaan 2.8

$$\delta_j = (\delta_{in_j} \times z_{out} (1 - z_{out})) \quad (2.8)$$

Keterangan :

δ_j = Nilai *error* unit tersembunyi

z_{out} = *Output* akhir pada *hidden layer*

δ_{in_j} = Nilai *error* pada *hidden layer*

Menghitung perbaikan pada bobot v_{ij} digunakan rumus pada Persamaan 2.9

$$\Delta v_{ij} = \alpha \delta_j x_i \quad (2.9)$$

Keterangan :

α = *Learning Rate*

δ_j = Nilai *error* unit tersembunyi

Δv_{ij} = Perubahan bobot unit *input* ke-i ke unit tersembunyi ke-j

x_i = Unit *input* ke-i untuk sebuah unit *input*, sinyal *input* dan sinyal *output* adalah sama yaitu x

Penyesuaian bobot di setiap unit *output* digunakan rumus pada Persamaan 2.10

$$W_{jk}(\text{baru}) = W_{jk}(\text{lama}) + \Delta W_{jk} \quad (2.10)$$

Keterangan :

$w_{jk}(\text{baru})$ = Bobot setelah diperbarui

$w_{jk}(\text{lama})$ = Bobot sebelum diperbarui

Δw_{jk} = Perubahan pada bobot w_{jk}

2.8 Bee Colony Optimization

Bee Colony Optimization merupakan algoritme kecerdasan yang meniru perilaku koloni lebah dalam mencari makan dengan cerdas. Dalam algoritme ini, setiap sumber madu dianggap sebagai solusi yang layak untuk masalah tersebut, di mana lebah-lebah dibagi menjadi *employee bee*, *onlooker bee*, dan *scout bee* sesuai dengan pembagian kerjanya dan dengan tiga tahap pencarian dalam algoritme tersebut. Lebah-lebah tersebut melakukan aktivitas yang berbeda sesuai dengan pembagian kerja masing-masing dan saling berbagi informasi untuk menyelesaikan proses pencarian masalah (Yang & Liu, 2023).

Berikut adalah tahapan yang dilakukan pada *Bee Colony Optimization* (Arfiani, Yuliansyah & Suratin, 2022).

1. Tentukan jumlah awal solusi potensial atau sumber makanan (x_{ij})
2. Mengevaluasi kualitas sumber makanan (f_i) dalam populasi, karena semakin banyak nektar pada sumber makanan, semakin besar kemungkinan lebah pencari akan memilihnya
3. Fase *Employee Bee* dengan menghitung (v_{ij}) yakni tetangga terdekat dari (x_{ij})
4. Fase *Onlooker Bee* yang akan mengevaluasi kembali dan menggabung sumber makanan dengan nilai probabilitas terbaik pada setiap cluster, dan menghasilkan sumber baru yang disebut dengan (x_{ij}). Langkah ini diulangi, namun sumber terbaik akan tetap disimpan di memori sebagai sumber terbaik pada iterasi tersebut
5. Pada fase *Scout Bee*, sumber makanan baru akan diproduksi selama jumlah sumber makanan belum mencapai batas siklus maksimum. Fase ini menghubungkan iterasi pertama dengan iterasi kedua dan seterusnya.
6. Sampai sumber yang paling cocok ditemukan, pengumpulan sumber akan dibandingkan dengan probabilitas kompatibilitas.

$$x_{ij} = l_{jk} + rand(0,1)(u_{jk} - l_{jk}) \quad (2.11)$$

Keterangan :

- SN = Jumlah sumber makanan
- D = Jumlah dimensi data
- K = Jumlah cluster
- k = {1,2 ..., K}
- i = {1,2, ..., SN}
- x_{ij} = i-sumber makanan awal dalam dimensi ke-j
- l_{jk} = nilai yang lebih rendah dari setiap nilai dalam dimensi-j untuk setiap cluster
- u_{jk} = nilai teratas setiap dimensi-j untuk setiap cluster
- Rand(0,1) = menghasilkan bilangan acak dengan distribusi biasa 0 samai 1

Memulai siklus iterasi sama dengan 1 hingga jumlah siklus maksimum. Untuk setiap solusi penghitungan lebah yang digunakan cara baru:

$$v_{ij} = z_{ij} + \phi_{ij}(z_{ij} - z_{kj}) \quad (2.12)$$

Keterangan :

- v_{ij} = kandidat posisi makanan baru berdasarkan posisi lama di memori.

\emptyset_{ij} = angka acak antara $\{-1,1\}$

Indeks acak dalam persamaan berfungsi untuk mengatur produksi sumber makanan di sekitar tetangga dan membandingkan dua posisi sumber makanan yang ditemukan oleh lebah. Langkah selanjutnya ialah menghitung nilai dan memilih sumber makanan terbaik di setiap cluster.

Setiap lebah pengamat akan menentukan solusi z_i berdasarkan p_i . Selanjutnya, hitung nilai v_i sesuai dengan langkah lebah pekerja. Membandingkan dua sumber makanan yang tersedia berdasarkan hasil probabilitas, solusi terbaik pada iterasi ini akan disimpan.

Berdasarkan hasil probabilitas, solusi terburuk yang tersisa ditemukan oleh lebah pengamat dan akan digantikan oleh solusi baru yang dibangkitkan secara acak.

$$Z_j^i = Z_j^{min} + rand(0,1)(Z_j^{max} - Z_j^{min}) \quad (2.13)$$

Keterangan :

Z^i = sumber makanan meninggalkan lebat pengamat dan $\{1,2,\dots\}$.

Ulangi siklus tersebut hingga jumlah siklus maksimum.

2.9 Alat dan Bahan

Pada penelitian ini akan menggunakan beberapa alat dan bahan yang tercantum dalam Tabel 2.2 dan 2.3

a. Alat

Alat penelitian merujuk pada segala jenis perangkat, instrumen, atau metode yang digunakan untuk mengumpulkan data atau informasi dalam suatu penelitian ilmiah. Alat yang digunakan pada penelitian klasifikasi jenis penyakit tanaman padi pada algoritme *Backpropagation* dan optimasi *Bee Colony Optimization* dapat dilihat pada Tabel 2.2

Tabel 2.2 Alat

Nama Alat	Keterangan
Perangkat	Laptop Asus X430FN_S430FN
Prosesor	Intel Core i5, RAM 12 GB, Windows 11
Bahasa Pemrograman	Python
Text Editor	Visual Studio Code, Google Colab
Perhitungan Manual	Microsoft Excel
Pembuat Laporan	Microsoft Word

b. Bahan

Bahan penelitian tertuju pada segala jenis materi atau sumber yang digunakan dalam suatu penelitian. Bahan yang digunakan dapat dilihat pada Tabel 2.3

Tabel 2.3 Bahan

Nama Bahan	Keterangan
Data gejala dan penyakit tanaman padi	Diperoleh dari artikel dan buku
Hasil kuesioner penyakit tanaman padi	Diperoleh dari petani

2.10 Evaluasi Kinerja

Tahap ini bertujuan untuk mengevaluasi kinerja algoritme yang telah dikembangkan. Pengujian dilakukan dengan mengukur tingkat akurasi klasifikasi algoritme yang menunjukkan seberapa dekat hasil pengujian algoritme dengan nilai sebenarnya (Herdiansah *et al.*, 2022).

2.10.1 Akurasi

Akurasi menunjukkan seberapa baik sistem dapat mengklasifikasikan data dengan benar. Nilai akurasi adalah rasio data yang diklasifikasikan dengan benar terhadap jumlah keseluruhan data. (Agustina, Magdalena & Pratiwi, 2022).

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} * 100\% \quad (2.14)$$

Keterangan :

True Positive (TP) : Jika data yang diprediksi bernilai positif dan sesuai dengan nilai actual (positif).

True Negative (TN) : Jika benar antara prediksi negatif dan aktualnya negatif

False Positive (FP) : Jika data yang diprediksi tidak sesuai dengan nilai actual.

False Negative (FN) : Jika yang diprediksi bernilai negatif dan aktualnya positif.

2.10.2 Presisi

Presisi adalah perbandingan nilai jumlah kategori data yang diklasifikasikan secara benar dengan jumlah total kategori data yang diklasifikasikan secara benar. (Agustina, Magdalena & Pratiwi, 2022).

$$Presisi = \frac{TP}{FP + TP} * 100\% \quad (2.15)$$

Keterangan :

True Positive (TP) : Jika data yang diprediksi bernilai positif dan sesuai dengan nilai actual (positif).

False Positive (FP) : Jika data yang diprediksi tidak sesuai dengan nilai actual.

2.10.3 Recall

Recall merupakan perbandingan antara item relevan yang dipilih dengan total jumlah keseluruhan item relevan yang ada (Argina, 2020).

$$Recall = \frac{TP}{FN + TP} * 100\% \quad (2.16)$$

Keterangan :

True Positive (TP) : Jika data yang diprediksi bernilai positif dan sesuai dengan nilai aktual (positif).

False Negative (FN) : Jika yang diprediksi bernilai negatif dan aktualnya positif.

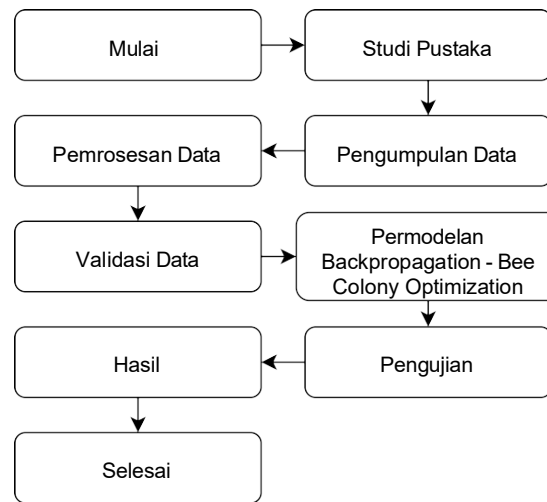
2.10.4 F-1 Score

F-1 Score merupakan metrik evaluasi temu kembali dengan menggabungkan *recall* dan *precision* (Habibi & Winar Cahyo, 2020).

$$F - 1 \text{ Score} = 2 \frac{(\text{Presisi} \times \text{Recall})}{(\text{Presisi} + \text{Recall})} \quad (2.17)$$

2.11 Diagram Alur Penelitian

Dalam penelitian ini dilakukan dalam beberapa tahapan yang dapat dilihat pada diagram alur penelitian pada Gambar 2.2



Gambar 2.2 Alur Penelitian

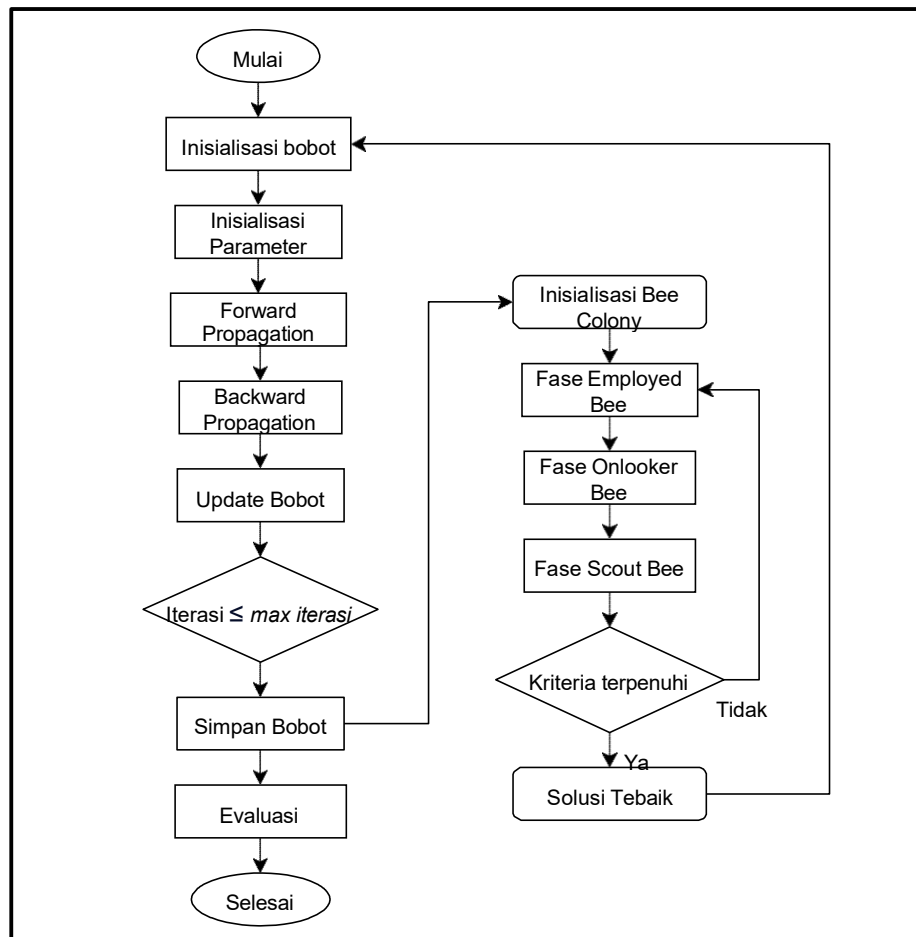
Berdasarkan diagram alur penelitian pada Gambar 2.2 tahapan yang dilakukan pada penelitian ini dapat dijelaskan sebagai berikut :

1. Memulai penelitian dengan melakukan studi pustaka untuk mengumpulkan dan mempelajari literatur atau referensi yang berkaitan dengan topik penelitian, seperti artikel, buku, jurnal, dan sumber-sumber lain.
2. Pada tahapan selanjutnya yaitu mengumpulkan data yang diperlukan untuk penelitian dengan melakukan wawancara dan memberikan kusioner kepada petani.
3. Setelah pengumpulan data, dilakukan pemrosesan data yang bertujuan untuk membersihkan data dengan menghilangkan noise atau kesalahan sehingga siap untuk di analisis lebih lanjut

4. Kemudian dilakukan validasi keakuratan dan keabsahan terhadap data yang telah melewati pemrosean data oleh pakar terkait
5. Selanjutnya, menerapkan algoritme *Backpropagation* dan *Bee Colony Optimization* untuk menghasilkan model yang sesuai dengan penelitian.
6. Pada tahap pengujian data, yaitu menguji hasil yang didapatkan dari permodelan data untuk mendapatkan hasil yang lebih baik
7. Setelah data diuji, hasil pengujian akan ditampilkan, hasil ini dapat berupa laporan, visualisasi data, dan *output* model yang telah diuji.

2.12 Alur Metode

Langkah Langkah dalam mengklasifikasi data dengan menggunakan algoritme *Backpropagation* dan *Bee Colony Optimization* dapat dilihat pada Gambar 2.3



Gambar 2.3 Alur Metode

Berdasarkan alur metode pada Gambar 2.3 tahapan yang dilakukan dapat dijelaskan sebagai berikut :

1. Proses dimulai dengan memuat data yang diperlukan ke dalam sistem yaitu data penyakit yang didapat dari petani

2. Model diinisialisasi dengan menentukan bobot random awal, parameter awal, seperti *hidden layer, epoch, learning rate* untuk memulai proses pelatihan.
3. *Forward Propagation* adalah proses yang digunakan untuk menghasilkan *output* sementara. Setiap unit *input* menerima sinyal dan meneruskannya ke unit tersembunyi, kemudian menghitung bobot sinyal *input*. menggunakan Persamaan 2.1.

$$z_{in_j} = 0,7000 \times (0 + 0,7000 \times 1 + 0,3430 \times 1 + 0,5518 \times 1 + 0,0866 \times 1 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0)$$

$$z_{in_j} = 1,1770$$

Hasil lengkap perhitungan bobot sinyal *input* dapat dilihat pada Tabel 2.4.

Tabel 2.4 Perhitungan Bobot Sinyal *Input*

Z_{in1}	Z_{in2}	Z_{in3}	Z_{in4}
1,3179	1,7659	1,8637	0,6830
4,0282	3,3092	3,0312	4,9538
1,2866	2,1892	0,9393	2,3642
0,5036	0,0640	0,8633	0,8793
1,7946	3,7888	0,4054	3,3458

Menghitung *output* pada *hidden layer* dengan fungsi aktivasi menggunakan Persamaan 2.2.

$$z_{out} = \frac{1}{1 + \exp(1,1770)}$$

$$z_{out} = 0,7644$$

Hasil lengkap perhitungan *output* pada *hidden layer* dengan fungsi aktivasi dapat dilihat pada Tabel 2.5

Tabel 2.5 Perhitungan *Output* pada *Hidden Layer*

z_{out1}	z_{out2}	z_{out3}	z_{out4}
0,7888	0,8539	0,8657	0,6644
0,9825	0,9647	0,9540	0,9930
0,7836	0,8993	0,7190	0,9141
0,6233	0,5160	0,7033	0,7067
0,8575	0,9779	0,6000	0,9660

Menghitung bobot sinyal *input* di unit *output* Y_k menggunakan Persamaan 2.3 & 2.4.

$$y_{in_k} = 0,0185 \times (0,7644 + 0,0185 \times 0,5884 + 0,0674 \times 0,5913 + 0,1816 \times 0,5001 + 0,1638 \times 0,0287)$$

$$y_{in_k} = 0,0203$$

$$y_k = \frac{1}{1 + \exp(0,0203)}$$

$$y_k = 0,5051$$

Hasil lengkap perhitungan bobot sinyal *input* di unit *output* Y_k dapat dilihat pada Tabel 2.6.

Tabel 2.6 Perhitungan Bobot Sinyal *Input*

y in	y out
0,0770	0,5192
0,0925	0,5231
0,0791	0,5198
0,0662	0,5165
0,0837	0,5209

4. *Backward Propagation* merupakan proses untuk menghitung kesalahan antara *output* yang dihasilkan dan *output* yang diharapkan, kemudian disebarkan kembali ke lapisan sebelumnya untuk memperbaiki bobot

Menghitung faktor δ unit *output* berdasarkan kesalahan di setiap unit *output* menggunakan Persamaan 2.5.

$$\delta_k = (0 - 0,5051) \times (0,5051 \times (1 - 0,5021))$$

$$\delta_k = -0,1263$$

Hasil lengkap perhitungan faktor δ unit *output* berdasarkan kesalahan di setiap unit *output* dapat dilihat pada Tabel 2.7.

Tabel 2.7 Perhitungan Faktor δ Unit *Output*

dk
-0,1296
0,1439
0,1698
-0,1290
0,1944

Menghitung perbaikan pada bobot w_{jk} yang menghubungkan *hidden layer* dan *input layer* menggunakan Persamaan 2.6.

$$\Delta w_{jk} = 1 \times -0,1263 \times (0,7644 + 0,5884 + 0,5913 + 0,5001)$$

$$\Delta w_{jk} = -0,3086$$

Hasil lengkap perhitungan perbaikan pada bobot w_{jk} yang menghubungkan *hidden layer* dan *input layer* dapat dilihat pada Tabel 2.8.

Tabel 2.8 Perhitungan Perbaikan pada Bobot

deltaW
-0,4113
0,5604
0,5630
-0,3289
0,6613

Menghitung faktor δ unit tersembunyi berdasarkan kesalahan di setiap unit tersembunyi z_j menggunakan Persamaan 2.7.

$$\delta_{in_j} = -1,1263 \times 0,0185$$

$$\delta_{in_j} = -0,0023$$

Hasil lengkap perhitungan faktor δ unit tersembunyi berdasarkan kesalahan di setiap unit tersembunyi z_j dapat dilihat pada Tabel 2.9.

Tabel 2.9 Perhitungan Faktor δ Unit Tersembunyi

dlnJ1	dlnJ2	dlnJ3	dlnJ4
-0,0074	-0,0092	-0,0258	-0,0228
0,0082	0,0102	0,0286	0,0254
0,0097	0,0121	0,0338	0,0299
-0,0074	-0,0092	-0,0257	-0,0227
0,0111	0,0138	0,0387	0,0343

Menghitung hasil bobot δ_j di *hidden layer* menggunakan Persamaan 2.8.

$$\delta_j = -0,0023 \times 0,7644 \times (1 - 0,7644)$$

$$\delta_j = -0,0004$$

Hasil lengkap perhitungan hasil bobot δ_j di *hidden layer* dapat dilihat pada Tabel 2.10.

Tabel 2.10 perhitungan hasil bobot δ_j

dj1	dj2	dj3	dj4
-0,0012	-0,0011	-0,0030	-0,0051
0,0001	0,0003	0,0013	0,0002
0,0016	0,0011	0,0068	0,0024
-0,0017	-0,0023	-0,0054	-0,0047
0,0014	0,0003	0,0093	0,0011

Menghitung perbaikan pada bobot v_{ij} menggunakan Persamaan 2.9.

$$\Delta v_{ij} = 1 \times -0,0004 \times 0$$

$$\Delta v_{ij} = 0$$

Hasil lengkap perhitungan perbaikan pada bobot v_{ij} dapat dilihat pada Tabel 2.11.

Tabel 2.11 Perbaikan pada Bobot v_{ij}

Δv_{10}	Δv_{20}	Δv_{30}	Δv_{40}	Δv_{11}	Δv_{21}	Δv_{31}	Δv_{41}
0,0000	-0,0011	-0,0030	-0,0051	-0,0012	0,0000	0,0000	0,0000
0,0000	0,0000	0,0000	0,0000	0,0000	0,0003	0,0013	0,0002
0,0016	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
0,0000	-0,0023	0,0000	-0,0047	-0,0017	0,0000	0,0000	0,0000
0,0014	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000

Δv_{12}	Δv_{22}	Δv_{32}	Δv_{42}	Δv_{13}	Δv_{23}	Δv_{33}	Δv_{43}
0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
0,0001	0,0003	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
0,0000	0,0000	0,0000	0,0024	0,0016	0,0011	0,0000	0,0000
0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
0,0000	0,0000	0,0093	0,0000	0,0000	0,0000	0,0093	0,0011

5. Update bobot memperbarui bobot model berdasarkan kesalahan perhitungan untuk meminimalkan kesalahan pada iterasi berikutnya.

Menghitung perbaikan pada bobot v_{ij} menggunakan Persamaan 2.10.

$$W_{jk}(\text{baru}) = 0,0185 + -0,3086$$

$$W_{jk}(\text{baru}) = -0,2901$$

Hasil lengkap perhitungan perbaikan pada bobot v_{ij} dapat dilihat pada Tabel 2.12.

Tabel 2.12 Perbaikan pada Bobot v_{ij}

Wbaru
-0,3541
0,6316
0,7620
-0,1527
0,7994

6. Mengulang proses pelatihan dengan algoritme *Backpropagation* hingga mencapai jumlah iterasi maksimum.
7. Algoritme *Bee Colony Optimization* dimulai dengan menginisialisasi populasi lebah, di mana *employee bee* mencari dan berbagi informasi sumber makanan, *onlooker bee* memilih sumber makanan berdasarkan probabilitas, kemudian solusi terbaik dari sumber makanan ditemukan, dan *scout bee* secara acak mencari sumber makanan baru untuk dieksplorasi lebih lanjut.
8. Membandingkan sumber makanan yang telah dikumpulkan dengan probabilitas kompatibilitas menggunakan Persamaan 2.11.

Jika nilai absolut

$$31,720 + 0,3 \times (31,720 - 12,906 < 1 = 1$$

Jika nilai absolut

$$31,720 + 0,3 \times (31,720 - 12,906 > 100 = 100$$

Jika tidak, hasilnya adalah absolut dari $31,720 + 0,3 \times (31,720 - 12,906)$

$$x_{ij} = 37,3644$$

Hasil lengkap perhitungan perbandingan sumber makanan yang telah dikumpulkan dengan probabilitas kompatibilitas dilihat pada Tabel 2.13.

Tabel 2.13 Perbandingan Sumber Makanan

w1	w2	w3	w4	w5
37,36442	35,15211	36,0647769	11,52489	40
1,536388	16,32753	36,4999232	28,1874	35
64,16316	59,47177	32,1907857	22,11779	37

Menghitung setiap solusi lebah baru menggunakan persamaan 2.12.

Jika nilai absolut

$$40 + 0,3 \times (40 - 15) < 1 = 1$$

Jika nilai absolut

$$40 + 0,3 \times (40 - 35) > 100 = 100$$

Jika tidak, hasilnya adalah absolut dari $40 + 0,3 \times (40 - 35)$

$$v_{ij} = 41$$

Hasil lengkap perhitungan setiap solusi lebah baru dilihat pada Tabel 2.14.

Tabel 2.14 Perhitungan Setiap Solusi Lebah Baru

w1	w2	w3	w4	w5
48,112835	40,7995	35,934233	6,526131	41
29,776996	5,24458	38,654492	31,22221	34
100	100	57,9434143	39,81202	67

Berdasarkan hasil probabilitas, solusi terburuk yang tersisa adalah lebah *onlooker* dan diganti dengan solusi baru yang diperoleh secara acak menggunakan Persamaan 2.13.

$$Z_j = 1 + 0,9 \times 88$$

$$Z_j = 80,2$$

Hasil lengkap perhitungan Berdasarkan hasil probabilitas, solusi terburuk yang tersisa adalah lebah *onlooker* dan diganti dengan solusi baru yang diperoleh secara acak dilihat pada Tabel 2.15

Tabel 2.15 Solusi Baru

w1	w2	w3	w4	w5
4,2	17	7	29,5	46
15,4	10,9	4,8	10,9	7,1
13	60,4	20,8	90,1	35,2

9. Memeriksa apakah kriteria penghentian telah terpenuhi, misalnya apakah solusi terbaik telah ditemukan atau iterasi maksimum telah tercapai.
10. Mengulang proses *Backpropagation* menggunakan solusi terbaik atau bobot yang didapatkan dari proses *Bee Colony Optimization*.
11. Mengevaluasi kinerja model dengan menggunakan metrik evaluasi seperti akurasi, presisi, dan *recall* yang didapat dari permodelan *Backpropagation* dan *Bee Colony Optimization*, selesai.