Identifikasi Gigitan Ular Menggunakan *Local Binary Pattern (LBP)* dan *AdaBoost*

Aldika Wicaksono[1], Dody Qori Utama[2], Adiwijaya[3]

[1,2,3]Fakultas Informatika, Universitas Telkom, Bandung
[1]aldikawicaksono@students.telkomuniversity.ac.id, [2]dodyqori@telkomuniversity.ac.id,
[3]adiwijaya@telkomuniversity.ac.id

Abstrak

Badan Kesehatan Dunia (WHO) memperkirakan lebih dari 5 juta orang di seluruh dunia terkena gigitan ular setiap tahunnya,dengan tingkat kematian mencapai lebih dari 100 ribu orang. Populasi ular seperti reptil lainnya akan berkembang biak secara optimal di iklim tropis dan lembab seperti di Indonesia, naiknya populasi ular berbanding lurus dengan naiknya kasus gigitan ular. Data di Indonesia menunjukan terjadinya 12.739 - 214.883 kasus gigitan ular dengan angka kematian 20 - 11.581 orang, sehingga Indonesia merupakan salah satu negara tropis dengan risiko tinggi terkena gigitan ular baik yang berbisa maupun tidak berbisa. Identifikasi penyebab bekas gigitan ular sangat penting dalam menolong korban, karena setelah terjadinya gigitan mempunyai perbedaan anatomi antara bekas gigitan ular yang berbisa dengan yang tidak berbisa. Penelitian ini mencoba membangun sistem identifikasi bekas gigitan ular yang berbisa dan tidak berbisa menggunakan teknologi berbasis image processing menggunakan metode *Local Binary Pattern* dan *AdaBoost,* dari hasil penelitian didapat nilai optimal akurasi sebesar 100% dari data latih dan 94% dari data uji dengan resolusi *pixel* 400 x 400.

Kata kunci : Gigitan ular, Local Binary Pattern, AdaBoost

Abstract

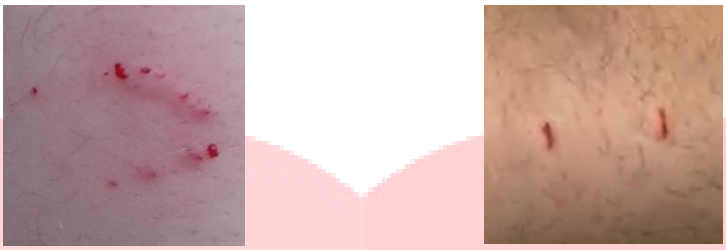
The World Health Organization (WHO) estimates that more than 5 million people worldwide are bitten by snakes every year, with a death rate reaching more than 100,000 people. Snake populations like other reptiles will reproduce optimally in tropical and humid climates such as in Indonesia, the increase in snake populations is directly proportional to the increase in snake bite cases. Data in Indonesia is estimated to have occurred 12,739 - 214,883 cases of snake bites with a death rate of 20 - 11,581 people, so that Indonesia is one of the tropical countries with a high risk of being bitten by both venomous and non-venomous snakes. Identifying the cause of snake bite marks is very important in helping the victim, because after the bite there is an anatomical difference between the bite marks of a venomous snake and a non-venomous one. This study tries to build a venomous and non-venomous snake bite mark identification system using image processing based technology using the *Local Binary Pattern* and *AdaBoost* methods, from the research results obtained the optimal value of accuracy of 100% from the training data and 94% from the test data with a *pixel* resolution of 400x400.

Keywords: Snake bite, Local Binary Pattern, AdaBoost

1. Pendahuluan

Indonesia sebagai salah satu negara tropis memiliki iklim yang hangat dan lembab merupakan habitat yang sangat sesuai untuk perkembangan ular, sehingga Indonesia juga merupakan salah satu negara berisiko tinggi terhadap gigitgan ular baik yang berbisa atau yang tidak berbisa. Bahkan menurut badan kesehatan dunia (WHO) diperkirakan sekitar 5,4 juta orang didunia terkena gigitan ular setiap tahunnya dan angka kematian 81 ribu hingga 138 ribu setiap tahunnya [1].

Di Indonesia dilaporkan terjadi kasus gigitan ular diperkirakan 12.739 – 214.883 kasus dengan tingkat kematian mencapai 20 – 11.581 jiwa [2]. Untuk menurunkan tingkat angka kematian akibat gigitan ular khususnya ular berbisa proses identifikasi bekas gigitan merupakan salah satu solusi[3]. Dan memberikan terapi pengobatan secara medis serta mengetahui jenis ular yang menggigit juga merupakan alternatif solusi yang lain [4].



(a) (b)

Gambar 1. Citra luka gigitan ular tidak berbisa[1] (a) Citra luka gigitan ular berbisa[2] (b)

Pada penelitian sebelumnya di tahun 2019, identifikasi gigitan ular menggunakan LBP (*Local Binary Pattern*) dan klasifikasi Naïve-Bayes didapatkan nilai akurasi 83,33% [5], sedangkan Rayiemas [10] melakukan klasifikasi gigitan ular menggunakan metode *Chain Code* dan *K-Nearest Neighbor* dengan nilai akurasi optimal 80%. Pada tahun sebelumnya yaitu 2015, penelitian tentang deteksi dan klasifikasi pada daun dengan metode *classifier AdaBoost* dan SVM dengan metode ekstraksi ciri *Haar like* didapatkan hasil terbaik nilai akurasi 84,23% dengan menggunakan Adaboost dan dengan menggunakan metode klasifikasi SVM akurasi yang didapatkan adalah 71%, sedangkan dengan menggunakan kombinasi kedua metode tersebut hasil yang didapatkan mencapai 51,68%. Tingkat akurasi yang relatif rendah tersebut disebabkan karena fitur yang digunakan hanyalah fitur tesktur, dimana fitur ini tidak cukup untuk membantu klasifikasi daun [6]. Berdasarkan latar belakang diatas, penulis akan membuat penelitian dengan metode yang berbeda tentang sistem identifikasi dan klasifikasi gigitan ular menggunakan metode LBP (*Local Binary Pattern*) dan *AdaBoost* dimana fitur yang digunakan adalah fitur jumlah gigitan, jarak gigitan dan tekstur gigitan ular dengan tujuan untuk meningkatkan akurasi dalam identifikasi gigitan ular.

Batasan masalah pada penelitian ini:

1. Dataset yang digunakan adalah data riil berupa gambar / foto baik untuk data latih maupun data uji yang didapatkan dari CommDIS Telkom University.

2. *Pixel* yang digunakan untuk data latih adalah 300x300, 400x400, dan 500x500.

3. Reproduksi dilakukan dengan cara masing-masing gambar dirotasi 0^o, 90^o, 180^o dan 270^o searah jarum jam.

4. Fokus penelitian ini mencari kombinasi terbaik dari ukuran *pixel* yang berbeda untuk data latih dan variasi jumlah data baik data latih maupun data uji, serta dilakukan klasifikasi menggunakan *AdaBoost* untuk mendapatkan klasifikasi gigitan ular berbisa dan gigitan ular tidak berbisa dengan nilai akurasi yang optimal.

2. Studi Terkait

Secara morfologi, ular tergolong hewan reptil dengan ciri khas dalam melumpuhkan mangsanya selalu menggunakan teknik lilitan badan dan gigitan. Dari struktur anatomi gigi, terlihat perbedaan mendasar antara ular berbisa yaitu adanya 2 gigi taring untuk menyuntikan racun sedangkan ular tidak berbisa tidak memiliki gigi taring [4]. Adanya struktur anatomi tersebut akan mempengaruhi bekas gigitannya. Ular berbisa akan meninggalkan 2 hingga 4 titik bekas gigitan taringnya [7].

Penelitian tentang klasifikasi kasus gigitan ular menggunakan gambar luka gigitan ular, pernah dilakukan oleh Nishioka [3]. Pada tahun 1995 dengan meneliti 42 gambar kasus gigitan ular dan menggunakan pengamatan secara visual. Hasil yang diperoleh adalah dapat mengenali 89% bekas luka gigitan ular berbisa dan 100% bekas luka gigitan ular tidak berbisa.

[1] J. Dorsett. Snake Vite. 2017. https://www.reptileforums.co.uk/forums/snakes/969098-snake-bites-pictures.html. [Diakses 30 September 2020].
[2] C. Fountain. FOX 46. 4 Mei 2017. http://www.fox46charlotte.com/news/local-news/man-shares-his-story-after-bit-by-cottonmouth-while-fishing-in-huntersville. [Diakses 30 September 2020].

Pada tahun 2015, Zaki [6] melakukan penelitian tentang deteksi dan klasifikasi pada daun menggunakan metode klasifikasi *AdaBoost* dan SVM serta menggunakan ekstraksi fitur *Haar like*, didapatkan akurasi terbaik 84,23% dengan menggunakan *AdaBoost*, dan dengan menggunakan metode klasifikasi SVM akurasi yang didapatkan adalah 71%, sedangkan dengan menggunakan kombinasi kedua metode tersebut hasil yang didapatkan mencapai 51,68%. Tingkat akurasi yang relatif rendah tersebut disebabkan karena fitur yang digunakan hanyalah fitur tesktur, dimana fitur ini tidak cukup untuk membantu klasifikasi daun.

Pada tahun 2019,dilakukan penelitian kasus gigitan ular menggunakan teknologi *Image Processing* dengan metode *Active Contour Model* dan *Support Vector Machine* (SVM) [8]. Didapatkan performansi yang cukup baik dan deteksi yang tepat dengan akurasi 100%, sedangkan Rayiemas [10] melakukan klasifikasi gigitan ular menggunakan metode *Chain Code* dan *K-Nearest Neighbor* dengan nilai akurasi optimal 80%. Pada tahun 2019 juga, Fathur melakukan penelitian Identifikasi gigitan ular menggunakan LBP (*Local Binary Pattern*) dan klasifikasi Naïve-Bayes didapatkan nilai akurasi 83,33% [5].

Pada penelitian lainnya mengenai *AdaBoost* pernah dilakukuan oleh Rani tahun 2011 [17] Klasifikasi Jenis Kelamin Berdasarkan Citra Wajah Menggunakan Kombinasi Algoritma *AdaBoost – Support Vector Machine* diperoleh akurasi sistem terbaik 86% jika dibandingkan dengan akurasi yang dihasilkan oleh single SVM *classifier*, tingkat akurasi yang dihasilkan oleh kombinasi *AdaBoost*-SVM ternyata tidak lebih baik.hal ini terjadi karena pada kasus klasifikasi jenis kelamin ini terdapat dilema antara akurasi dan *diversity*. Jika kombinasi klasifier yang akurat namun memiliki *diversity* yang rendah maka akan menyebabkan performa algoritma *AdaBoost* menjadi rendah

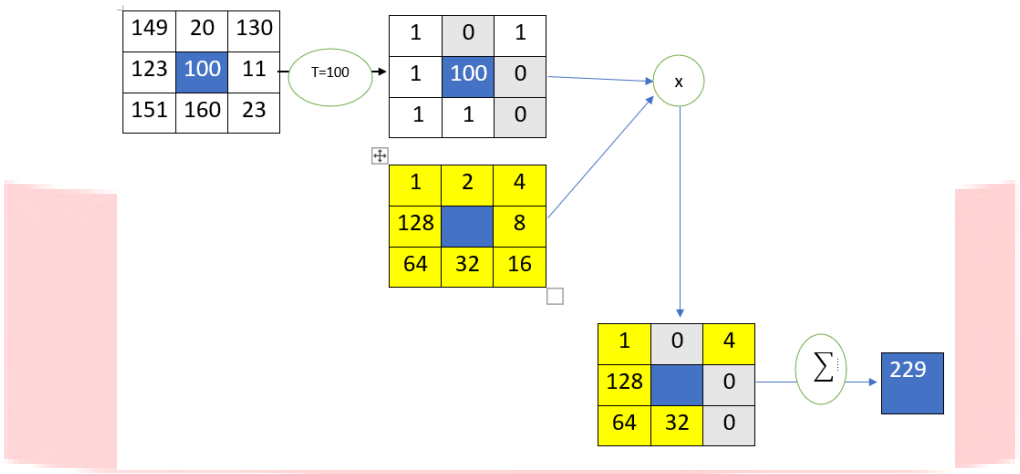
Local Binary Pattern (LBP) merupakan metode analisis yang menggunakan model statistika dan terstruktur. Standar operasi LBP bekerja pada blok piksel 3x3 dengan nilai tengah pada blok menjadi nilai threshold terhadap blok tetangganya pada citra *grayscale*[9]. Apabila nilai bernilai sama atau lebih besar maka akan diberi nilai 1. Selain itu, akan diberi nilai 0.

Salah satu jenis algoritma yang digunakan untuk melakukan pelatihan data dalam mengklasifikasikan bentuk citra gigitan ular adalah *boosting*. Bentuk boosting yang banyak digunakan adalah adaptive boosting, disingkat *AdaBoost*.Menurut Lienhart[11] *boosting* adalah sekumpulan *classifier* seperti sebuah pohon keputusan dimana setiap tingkat keputusan dilatih untuk mendeteksi keseluruhan objek dan menolak objek yang tidak sesuai kriteria. Boosting merupakan teknik yang sangat baik untuk mengkombinasikan banyak *classifier* membentuk kesatuan yang hasil akhirnya lebih baik dibanding hasil tiap classifier dasar. Secara umum, penggunaan Algoritma *Adaboost* sebagai klasifier memiliki beberapa keunggulan, terutama jika data yang digunakan sederhana. Artinya kurang dari (3) tiga parameter karena *Adaboost* langsung mengubah data *Weak Classifier* menjadi *Strong Classifier*. Selain itu, penggunaan *Adaboost* serbaguna karena berbagai macam data, berupa gambar, video dan suara bisa digunakan. Dari sisi tingkat kesulitan pemograman, penggunaan *Adaboost* relatif lebih mudah**.** Menurut Freund dan Schapire [12] dan Zhi-Hua[13] performa *boosting* dapat menghasilkan klasifikasi yang bagus, meskipun tiap *classifier* dasarnya tidak sebagus algoritma random dan satuan *classifier* tersebut disebut *weak learner*.

3. Sistem Yang Dibangun

3.1. Ekstraksi Local Binary Pattern (LBP)

Local Binary Pattern (LBP) merupakan salah satu jenis ekstraksi fitur yang didefinisikan sebagai ukuran tekstur *grayscale* yang berasal dari sekitarnya. LBP telah banyak digunakan dalam berbagai aplikasi penelitian citra karena memiliki keunggulan dalam analisis ekstraksi fitur terutama dalam deskripsi tekstur dan efisiensi komputasi [15]. Cara kerja LBP adalah membandingkan nilai biner *pixel* pusat citra dan pixel sekitarnya. Dengan menggunakan blok *pixel* 3x3 dan threshold sebagai nilai tengah *pixel* dimana *pixel* di pusat citra akan dikurangi dengan *pixel* sekitarnya. Jika hasilnya lebih besar atau sama dengan 0 akan diberi nilai 1, dan sebaliknya jika hasilnya kurang dari 0 akan diberi nilai 0. Setelah itu 8 nilai pada blok 3x3 disusun searah jarum jam. Kemudian diubah dalam bentuk desimal sebagai pengganti nilai *pixel* pada pusat citra [16].



Gambar 2 *Local Binary Pattern*

Secara umum, LBP membandingkan setiap tetangga dengan cara sebagai berikut.

$$LBP_{p_r} = \sum_{p=0}^{p-1} s(g_p - g_c)\, 2^p, s(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$$

g_c merupakan nilai tengah *pixel*, g_p merupakan nilai dari tetangga, p merupakan jumlah tetangga yang berhubungan, dan r merupakan radius dari tetangga. Setelah LBP terindentifikasi, LBP akan menghasilkan karakteristik gambar dalam histogram, yang merepresentasikan sebagai matriks berbentuk 1×n.

3.2. *Classifier AdaBoost*

AdaBoost adalah singkatan dari *adaptive boosting*, algoritma *AdaBoost* selain dapat berfungsi menyesuaikan diri (*adaptive*) terhadap data dan metode *classifier* lain juga sebagai *boosting* yang fungsinya dapat menurunkan tingkat kesalahan dari *classifier* lemah serta dapat meningkatkan tingkat akurasi dari setiap algoritma pembelajaran yang digunakan [12].

Klasifikasi adalah salah satu *task* analisis data objek dalam bentuk model untuk kategori / kelas yang sudah didefinisikan. Proses klasifikasi terdiri dari 2 tahap yaitu tahap pertama adalah tahap *training* / latih dan tahap kedua adalah tahap *testing* / uji. Pada tahap pertama, persamaan algoritma klasifikasi akan membentuk *classifier* data latih dari sekumpulan data latih yang digunakan dan disimpan dalam bentuk model. Untuk tahap berikutnya, model yang dihasilkan dari tahap pertama akan digunakan untuk mengklasifikasi data uji. Akurasi dari sebuah *classifier* untuk sekumpulan data uji yang diberikan merupakan persentase dari data-data uji yang diklasifikasikan dengan benar oleh *classifier*. Jika akurasi dari *classifier* dianggap cukup baik, *classifier* dapat digunakan untuk mengklasifikasikan data selanjutnya yang belum diketahui label kelasnya.

Contoh kasus klasifikasi dengan dua kategori, dengan kode variabel output $Y \in \{-1,1\}$. Setelah diberi vektor variabel yang diprediksi X maka *classifier* $G(X)$ akan memprediksi dengan menghasilkan dua nilai yaitu nilai {-1,1}. Untuk algoritma *AdaBoost* akan dihasilkan sebuah himpunan berurutan dari beberapa *classifier* lemah $G_m\,(x), m = 1,2,\ldots,M$. Untuk menghasilkan prediksi akhir, prediksi semua *classifier* akan dikombinasikan dalam voting berbobot. Pada tahap *boosting* selalu dilakukan modifikasi data dengan bobot $w_1, w_2, \ldots, w_N$ untuk setiap analisis data latih $(x_i, y_i), i = 1,2,\ldots,N$. Untuk iterasi awal, semua bobot diberi nilai $w_i = 1/N$, sehingga langkah awal akan melatih *classifier* dengan data dan cara yang umum. Iterasi selanjutnya $m = 2,3,\ldots,M$ akan dilakukan modifikasi bobot dan algoritma klasifikasi diaplikasikan pada data *training*. Pada Langkah iterasi ke m, akan dinaikkan bobotnya untuk data yang salah klasifikasinya oleh *classifier* $G_{m-1}(x)$ dan sebaliknya untuk data yang sudah benar bobot akan diturunkan. Dengan cara iterasi seperti ini otomatis untuk data yang sulit diklasifikasi akan mendapat pengaruh bobot yang meningkat sehingga pada iterasi berikutnya *classifier* akan dipaksa untuk fokus pada data latih yang salah akibat *classifier* sebelumnya.

Persamaan Algoritma *AdaBoost* disusun sebagai berikut:

Data citra diberikan label $(x_1,y_1),\ldots,(x_n,y_n)$ dimana $y_i = 0,1$ untuk data negatif dan positif berturut-turut.

Untuk setiap citra latih, diberi koordinat(x,y) dengan y = 0 untuk citra negatif, dan y = 1 untuk citra positif.

Inisialisasikan bobot $W_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ untuk $y_i = 0,1$ berturut turut, dimana m dan l adalah jumlah negatif dan positif berturut-turut.

Setiap citra diberi bobot awal yang sama, $\frac{1}{2m}$ untuk citra negatif, dan $\frac{1}{2l}$ untuk citra positif. Dimana m adalah jumlah total citra negatif dan l adalah total citra positif yang digunakan untuk proses training.

Untuk t = 1,…,T ;

1. Normalisasikan bobot,
$$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} W_{t,j}}$$
2. Untuk setiap fitur j , latih sebuah classifier h_j yang dibatasi agar menggunakan sebuah fitur tunggal. Kesalahan dievaluasi sehubungan dengan
$$w_t \in_j = \sum_{i=1} w_i |h_j(x_i) - y_i|$$
3. Pilih classifier h_t dengan kesalahan terendah $\in_t$.
4. Perbaharui bobot :
$$w_{t+1,i} = w_t, i\beta_t^{1-e_i}$$
dimana $e_i = 0$ jika data x_i diklasifikasi dengan benar, $e_i = 1$ jika sebaliknya, dan $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.

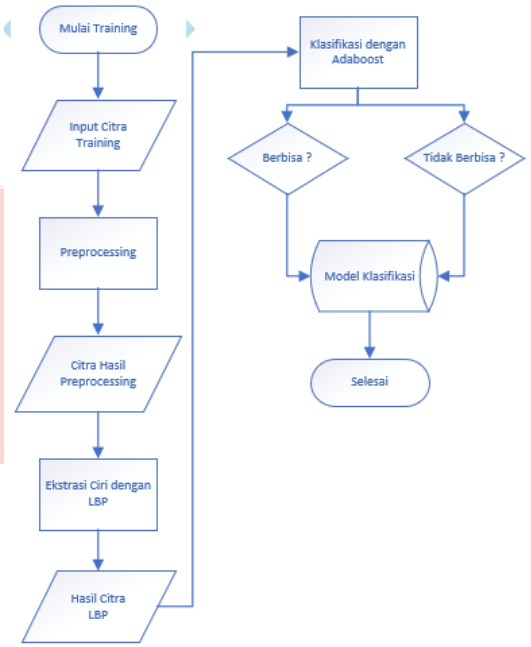
Untuk menentukan satu *classifier* terbaik diperlukan beberapa iterasi perhitungan seperti menghitung *error* dan *update* bobot. Hasil perolehan fitur yang memiliki *fitness function* paling optimal atau *error* yang paling rendah maka dia yang akan masuk ke *classifier* pertama.

$$H(x) = sign \sum_{t=1}^{T} \alpha_t h_t(x)$$

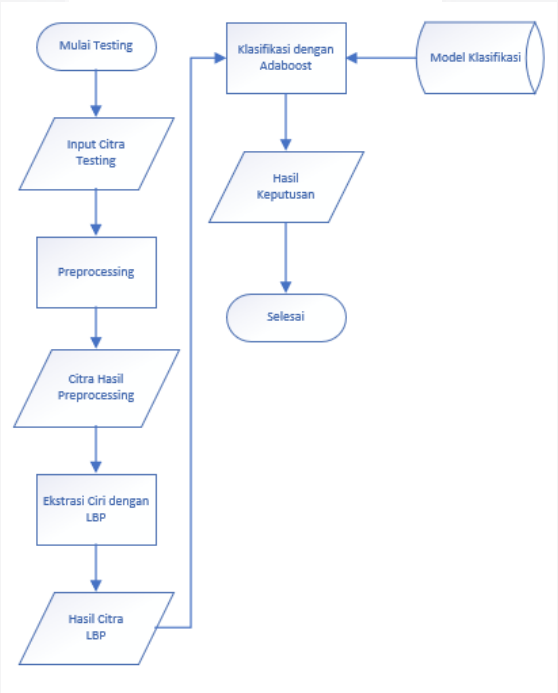
Hal ini dilakukan secara berulang selama iterasi tertentu hingga didapatkan *weak classifier* terpilih untuk menentukan keberadaan objek yang ditempatkan pada masing-masing *stage* yang telah terbentuk dengan metode *AdaBoost*. Semua persamaan ini diambil dari sumber referensi [14].

3.3. Sistem blok data latih dan data uji

Sistem deteksi dan klasifikasi yang dibangun pada penelitian ini dibagi dua bagian, yaitu proses latih (*training*) dan proses pengujian (*testing*). Pada proses latih akan dihasilkan sebuah model klasifikasi yang berisi data gigitan ular berbisa atau tidak berbisa, sedangkan pada proses pengujian akan dihasilkan hasil dari klasifikasi gigitan ular yaitu berupa label gigitan ular berbisa atau tidak berbisa pada setiap citra. Kemudian, hasil klasifikasi pengujian akan dibandingkan dengan model klasifikasi *training*, sehingga bisa diambil kesimpulan apakah hasil klasifikasi pengujian berasal dari gigitan ular berbisa atau tidak berbisa. Berikut ini adalah diagram alur dari tahap pembangunan model klasifikasi pada proses latih (*training)* dan uji (*testing*).



Gambar 3 Diagram Alur Training Sistem



Gambar 4 Diagram Alur Testing Sistem

Gambar 1 merupakan proses latih yang dimulai dari *input* citra *training* dan akan dihasilkan model klasifikasi yang akan menjadi *database* yang nantinya menjadi pembanding yang dibutuhkan untuk proses uji.

Pada proses *training* hal yang pertama dilakukan adalah melakukan *input* citra proses yang berisi gambar gigitan ular baik yang berbisa maupun yang tidak berbisa. Selanjutnya dilakukan *pre-processing* dimana hasil yang dikeluarkan adalah sebuah citra yang telah diubah ke dalam citra keabuan (*grayscale*). Setelah itu, citra keabuan akan di segmentasi sehingga mendapatkan gigitan ular yang berbisa atau tidak berbisa. Tujuan dari segmentasi ini sendiri adalah memudahkan pada saat klasifikasi. Selanjutnya akan dilakukan proses ekstraksi

ciri dengan menggunakan *Local Binary Pattern*, yaitu ekstraksi ciri yang menggunakan metode *Imbinerisasi* dengan hasil akhir berupa 0 dan 1 pada setiap *pixel* citra dan hasil ini akan diproses menggunakan metode klasifikasi *AdaBoost.*

Hasil dari proses tersebut berupa model klasifikasi yang berisi *database*, baik berupa database gigitan ular berbisa maupun yang tidak berbisa, selanjutnya akan digunakan sebagai pembanding pada proses pengujian (*testing).*

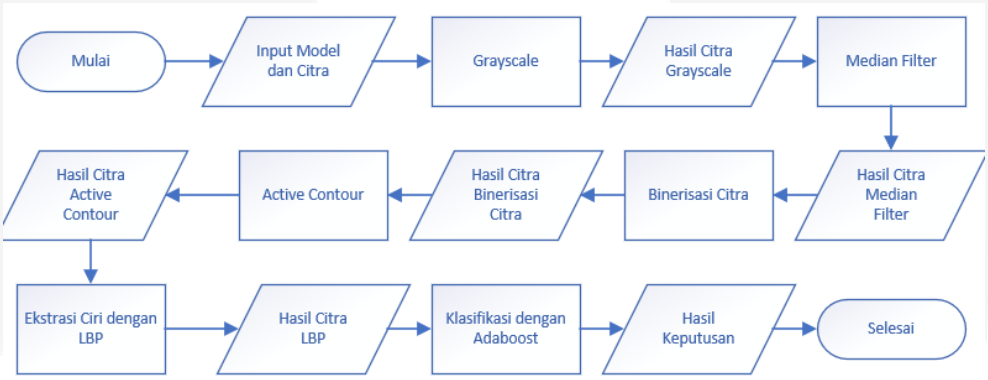
Pada gambar 2 merupakan alur pengujian, alur tersebut memiliki alur yang hampir sama dengan alur proses latih (gambar 1). Perbedaan mendasar adalah jumlah data yang digunakan antara alur proses latih dan alur pengujian. Hal lain yang berbeda adalah adanya proses perbandingan pada saat setelah klasifikasi dengan *Adaboost* dan hasil akhir alur pengujian berupa keputusan apakah citra tersebut berasal dari gigitan ular berbisa atau tidak berbisa.

3.4. Blok Pengolahan Citra

Data yang digunakan dalam pengembangan model klasifikasi gigitan ular adalah dua buah *dataset* berupa foto gigitan ular. Berikut adalah karekterisitik dataset:

1. Jumlah *dataset* adalah : 20 gambar, terdiri dari 13 gambar gigitan ular berbisa dan 7 gambar gigitan ular tidak berbisa.
2. Jenis citra yang ada pada setiap citra adalah jenis citra RGB.
3. Ada dua (2) kelas citra gigitan ular yaitu : gigitan ular berbisa dan gigitan ular tidak berbisa.
4. Ekstensi citra adalah JPG.

Diagram alur tahapan preprocessing digambarkan pada Gambar 3



Gambar 5 Diagram Alur Preprocessing

1. Semua input citra akan diubah terlebih dahulu menjadi warna keabuan. Proses ini disebut *Grayscaling*. *Grayscaling* ini mengubah citra RGB menjadi keabuan dengan persamaan $y = 0.299 * R + 0.587 * G + 0.114 * B$. Ini berarti dengan perbandingan R:G:B adalah 2:4:1
2. Hasil dari *Grayscaling* akan diberi *filter* menggunakan *median filter* dengan ukuran jendela 3x3 supaya dihasilkan gambar yang lebih halus dan menghilangkan *noise*.
3. Proses Binerisasi digunakan untuk mengubah nilai *pixel* pada citra menjadi 0 atau 1.
4. Setelah proses Binerisasi, akan dilakukan metode *Active Contour* dengan iterasi 500 dan inisialisasi border 25 yang digunakan untuk membantu melakukan deteksi objek dan mengenali objek berupa titik bekas gigitan. Setelah tahapan *preprocessing*, dilakukan proses tambahan pada *processing* khusus citra dengan label ular berbisa saja, sedangkan label ular tidak berbisa tidak perlu dilakukan proses tambahan dan akhirnya disimpan dalam bentuk (.JPG).
5. Langkah terakhir ekstraksi ciri menggunakan LBP dengan menampilkan histogram LBP dan kemudian diklasifikasikan menggunakan *AdaBoost* menghasilkan keputusan apakah citra tersebut berbisa atau tidak berbisa.

Untuk mengukur kinerja sistem dari model, dibutuhkan *Confusion Matrix* yang dapat merepresentasikan performansi sistem yang telah dibangun. Cara kerja *Confusion Matrix* adalah untuk membandingkan hasil dari prediksi klasifikasi dengan hasil seharusnya yang didapat dari klasifikasi.

Parameter		**Kelas Prediksi**	
		True	***False***
Kelas Aktual	***True***	*True Positive* (TP)	*False Positive* (FP)
	False	*True Negative* (TN)	*False Negative* (FN)

Ada empat istilah yang digunakan sebagai gambaran hasil proses klasifikasi pada *confusion matrix*, yaitu :

1. TP (*True Positive*) yaitu jumlah data positif yang terklasifikasi dengan benar oleh sistem. Dalam kasus ini adalah jumlah gigitan ular berbisa yang terklasifikasi dengan benar sebagai gigitan ular berbisa.
2. TN (*True Negative*) yaitu jumlah data negatif yang terklasifikasi dengan benar oleh sistem. Dalam kasus ini adalah jumlah gigitan ular tidak berbisa yang terklasifikasi dengan benar sebagai gigitan ular tidak berbisa
3. FN (*False Negative*) yaitu jumlah data negatif namun terklasifikasi salah oleh sistem. Dalam kasus ini adalah jumlah gigitan ular tidak berbisa yang terklasifikasi dengan salah sebagai gigitan ular tidak berbisa.
4. FP (*False Positive*) yaitu jumlah data positif namun terklasifikasi salah oleh sistem. Dalam kasus ini adalah jumlah gigitan ular berbisa yang terklasifikasi dengan salah sebagai gigitan ular berbisa.

Dengan menggunakan tabel *Confusion Matrix*, maka dapat dihasilkan metrik *Accuracy*, *Precision*, *Recall* dan *F1 Score.* Berikut adalah penjelasan dari masing-masing metrik.

1. *Accuracy*
   Accuracy merepresentasikan persentase jumlah prediksi akurat yang diberikan oleh model. Berikut adalah rumus untuk menghitung *Accuracy*
   $$Accuracy = \frac{TP + TN}{TP+FP+FN+TN}$$
2. *Precision*
   Precision merepresentasikan jumlah data kategori positif yang diklasifikasikan secara benar dibagi dengan total data yang diklasifikasi positif. Berikut adalah rumus untuk menghitung *Precision*.
   $$Precision = \frac{TP}{TP+FP}$$
3. *Recall*
   Recall merepresentasikan berapa persen data kategori positif yang terklasifikasikan dengan benar oleh sistem. Berikut adalah rumus untuk menghitung *Recall*.
   $$Recall = \frac{TP}{TP+FN}$$
4. *F1 Score*
   F1 Score merupakan perbandingan rata-rata presisi dan recall yang dibobotkan. Berikut adalah rumus untuk menghitung *F1 Score.*
   $$F1\ Score = \frac{2\ x\ Precison\ x\ Recall}{Precision+Recall}$$

4. Evaluasi

4.1. *Dataset*

Pengujian sistem dilakukan dengan menggunakan data sebagai berikut :

1. *Dataset* riil gambar citra sebagai **data latih** sebanyak 20 citra dengan komposisi 13 citra gigitan ular berbisa dan 7 citra gigitan ular tidak berbisa.
2. *Dataset* riil gambar citra sebagai **data uji** sebanyak 51 citra dengan komposisi 48 citra gigitan ular berbisa dan 3 citra gigitan ular tidak berbisa.

3. *Pixel* yang digunakan untuk data latih 300x300, 400x400 dan 500x500.
4. Reproduksi data dilakukan dengan rotasi 0^o , 90^o ,180^o dan 270^o searah jarum jam, sehingga didapatkan jumlah data latih reproduksi sebanyak 80 citra dan data uji reproduksi sebanyak 204 citra.

4.2. Hasil Pengujian

Ada dua skenario penelitian yang dilakukan untuk menguji kinerja sistem dan mencari nilai optimal dari kombinasi yang dibuat, yaitu :

1) Pengujian kinerja sistem dengan beberapa nilai *pixel* dari *dataset* yang didapat untuk pengujian data latih. Hasil pengujian sebagai berikut.

a. Pengujian Kinerja Sistem Dengan *Pixel* 300 x 300

Tabel 1. Hasil pengujian dengan Pixel 300 x 300

Learning Cycle	Nilai Akurasi		Nilai Presisi		Nilai Recall		Nilai F1 Score	
Load Model	Data Latih (20)	Data Uji (51)	Data Latih (20)	Data Uji (51)	Data Latih (20)	Data Uji (51)	Data Latih (20)	Data Uji (51)
1	85.00%	72.55%	0.81	0.93	1.00	0.77	0.90	0.84
2	85.00%	72.55%	0.81	0.93	1.00	0.77	0.90	0.84
3	95.00%	35.29%	1.00	1.00	0.92	0.31	0.96	0.48
4	85.00%	72.55%	0.81	0.93	1.00	0.77	0.90	0.84
5	100%	45.10%	1.00	1.00	1.00	0.42	1.00	0.59
6	100%	50.98%	1.00	1.00	1.00	0.48	1.00	0.65
7	100%	45.10%	1.00	1.00	1.00	0.42	1.00	0.59
8	100%	74.51%	1.00	0.93	1.00	0.79	1.00	0.85
9	100%	47.06%	1.00	1.00	1.00	0.44	1.00	0.61
10	100%	68.63%	1.00	0.92	1.00	0.73	1.00	0.81
11	100%	47.06%	1.00	0.89	1.00	0.50	1.00	0.64
12	100%	45.10%	1.00	0.88	1.00	0.48	1.00	0.62
13	100%	47.06%	1.00	0.89	1.00	0.50	1.00	0.64
14	100%	45.10%	1.00	0.88	1.00	0.48	1.00	0.62
15	100%	45.10%	1.00	0.88	1.00	0.48	1.00	0.62

b. Pengujian Kinerja Sistem Dengan *Pixel* 400 x 400

Tabel 2. Hasil pengujian dengan Pixel 400 x 400

Learning Cycle	Nilai Akurasi		Nilai Presisi		Nilai Recall		Nilai F1 Score	
Load Model	Data Latih (20)	Data Uji (51)	Data Latih (20)	Data Uji (51)	Data Latih (20)	Data Uji (51)	Data Latih (20)	Data Uji (51)
1	80.00%	94.11%	0.76	0.94	1.00	1.00	0.87	0.97
2	70.00%	29.41%	0.89	1.00	0.62	0.25	0.73	0.40
3	90.00%	94.11%	0.87	0.94	1.00	1.00	0.93	0.97
4	85.00%	68.62%	0.81	0.92	1.00	0.73	0.90	0.81
5	100%	92.15%	1.00	0.94	1.00	0.98	1.00	0.96
6	100%	92.15%	1.00	0.94	1.00	0.98	1.00	0.96
7	100%	68.62%	1.00	0.92	1.00	0.73	1.00	0.81
8	100%	92.15%	1.00	0.94	1.00	0.98	1.00	0.96
9	100%	56.86%	1.00	0.91	1.00	0.60	1.00	0.73
10	100%	70.59%	1.00	0.92	1.00	0.75	1.00	0.83
11	100%	58.82%	1.00	0.91	1.00	0.63	1.00	0.74
12	100%	58.82%	1.00	0.91	1.00	0.63	1.00	0.74
13	100%	56.86%	1.00	0.91	1.00	0.60	1.00	0.73
14	100%	58.82%	1.00	0.91	1.00	0.63	1.00	0.74
15	100%	56.86%	1.00	0.91	1.00	0.60	1.00	0.73

c. Pengujian Kinerja Sistem Dengan *Pixel* 500 x 500

Tabel 3. Hasil pengujian dengan Pixel 500 x 500

Learning Cycle	Nilai Akurasi		Nilai Presisi		Nilai Recall		Nilai F1 Score	
Load Model	Data Latih (20)	Data Uji (51)	Data Latih (20)	Data Uji (51)	Data Latih (20)	Data Uji (51)	Data Latih (20)	Data Uji (51)
1	80.00%	94.12%	0.76	0.94	1.00	1.00	0.87	0.97
2	70.00%	25.49%	0.89	1.00	0.62	0.21	0.73	0.34
3	90.00%	62.75%	1.00	1.00	0.85	0.60	0.92	0.75
4	85.00%	64.71%	0.81	1.00	1.00	0.63	0.90	0.77
5	100%	76.47%	1.00	1.00	1.00	0.75	1.00	0.86
6	100%	58.82%	1.00	1.00	1.00	0.56	1.00	0.72
7	100%	58.82%	1.00	1.00	1.00	0.56	1.00	0.72
8	100%	62.75%	1.00	1.00	1.00	0.60	1.00	0.75
9	100%	58.82%	1.00	1.00	1.00	0.56	1.00	0.72
10	100%	62.75%	1.00	1.00	1.00	0.60	1.00	0.75
11	100%	76.47%	1.00	1.00	1.00	0.75	1.00	0.86
12	100%	62.75%	1.00	1.00	1.00	0.60	1.00	0.75
13	100%	80.39%	1.00	1.00	1.00	0.79	1.00	0.88
14	100%	62.75%	1.00	1.00	1.00	0.60	1.00	0.75
15	100%	62.75%	1.00	1.00	1.00	0.60	1.00	0.75

Dari tabel 1,2 dan 3 diperoleh hasil pengujian kinerja sistem dengan nilai akurasi, presisi, *recall*, dan *F1 Score* yang tertinggi adalah pada *pixel* 400 x 400 baik untuk data latih dan data uji. Untuk data latih, nilai akurasi, presisi, *recall, dan F1 Score* 100% didapat pada saat *learning cycle* ke-5,6 ,7, dan seterusnya, sedangkan untuk data uji, nilai tertinggi terjadi pada saat *learning cycle* ke-3 dengan nilai akurasi 94%, nilai presisi 0,94, nilai *recall* 1, dan nilai *F1 Score* 0,97. Sementara untuk *pixel* 300x300 nilai tingkat akurasi tertinggi hanya 74,51% dengan presisi 1,00 *recall* 0,77 dan *F1 score* 0,84, serta untuk *pixel* 500x500 nilai tingkat akurasi tertinggi 94,12% presisi 1,00 *recall* 1,00 dan *F1 score* 0,97. Tetapi nilai tersebut terjadi pada saat nilai tingkat akurasi data latih belum sempurna, yaitu hanya 80%.

2) Pengujian kinerja sistem dengan kombinasi jumlah data baik data latih dan data uji menggunakan *pixel* 400 x400. Kombinasi yang dilakukan sebagai berikut :

a. Pengujian Kinerja Sistem dengan kombinasi jumlah data latih normal dan data uji normal

Tabel 4. Hasil Pengujian Kombinasi Data Latih Normal Dan Data Uji Normal

Learning Cycle	Nilai Akurasi		Nilai Precison		Nilai Recall		Nilai F1 Score	
Load Model	Data Latih (20)	Data Uji (51)	Data Latih (20)	Data Uji (51)	Data Latih (20)	Data Uji (51)	Data Latih (20)	Data Uji (51)
1	80.00%	94.11%	0.76	0.94	1.00	1.00	0.87	0.97
2	70.00%	29.41%	0.89	1.00	0.62	0.25	0.73	0.40
3	90.00%	94.11%	0.87	0.94	1.00	1.00	0.93	0.97
4	85.00%	68.62%	0.81	0.92	1.00	0.73	0.90	0.81
5	100%	92.15%	1.00	0.94	1.00	0.98	1.00	0.96
6	100%	92.15%	1.00	0.94	1.00	0.98	1.00	0.96
7	100%	68.62%	1.00	0.92	1.00	0.73	1.00	0.81
8	100%	92.15%	1.00	0.94	1.00	0.98	1.00	0.96
9	100%	56.86%	1.00	0.91	1.00	0.60	1.00	0.73
10	100%	70.59%	1.00	0.92	1.00	0.75	1.00	0.83
11	100%	58.82%	1.00	0.91	1.00	0.63	1.00	0.74
12	100%	58.82%	1.00	0.91	1.00	0.63	1.00	0.74
13	100%	56.86%	1.00	0.91	1.00	0.60	1.00	0.73
14	100%	58.82%	1.00	0.91	1.00	0.63	1.00	0.74
15	100%	56.86%	1.00	0.91	1.00	0.60	1.00	0.73

Dari tabel 4, diperoleh nilai tertinggi baik untuk akurasi, presisi, *recall*,dan *F1 Score* untuk data latih sebesar 100% pada *learning cycle* ke-5, 6, 7, dan seterusnya. Sedangkan nilai data uji tertinggi untuk nilai akurasi sebesar 94%, nilai presisi 0,94, nilai *recall* 1, dan nilai *F1 score* 0,97 pada *learning cycle* ke 1 dan 3.

b. Pengujian Kinerja Sistem dengan kombinasi jumlah data latih reproduksi dan data uji normal

Tabel 5. Hasil Pengujian Kombinasi Data Latih Reproduksi Dan Data Uji Normal

Learning Cycle	Nilai Akurasi		Nilai Precison		Nilai Recall		Nilai F1 Score	
Load Model	Data Latih (80)	Data Uji (51)	Data Latih (80)	Data Uji (51)	Data Latih (80)	Data Uji (51)	Data Latih (80)	Data Uji (51)
1	80.00%	94.12%	0.76	0.94	1.00	1.00	0.87	0.97
2	80.00%	94.12%	0.76	0.94	1.00	1.00	0.87	0.97
3	85.00%	84.31%	0.81	0.93	1.00	0.90	0.90	0.91
4	80.00%	94.12%	0.76	0.94	1.00	1.00	0.87	0.97
5	93.75%	90.20%	0.96	0.94	0.94	0.96	0.95	0.95
6	91.25%	94.12%	0.89	0.94	0.98	1.00	0.94	0.97
7	92.50%	94.12%	0.90	0.94	1.00	1.00	0.95	0.97
8	93.75%	62.75%	1.00	0.97	0.90	0.63	0.95	0.76
9	100.00%	66.67%	1.00	0.97	1.00	0.67	1.00	0.79
10	100.00%	66.67%	1.00	0.97	1.00	0.67	1.00	0.79
11	100.00%	66.67%	1.00	0.97	1.00	0.67	1.00	0.79
12	100.00%	94.12%	1.00	0.94	1.00	1.00	1.00	0.97
13	100.00%	92.16%	1.00	0.96	1.00	0.96	1.00	0.96
14	100.00%	90.20%	1.00	0.98	1.00	0.92	1.00	0.95
15	100.00%	92.16%	1.00	0.96	1.00	0.96	1.00	0.96

Dari tabel 5, diperoleh nilai tertinggi baik untuk akurasi,presisi,*recall*,dan *F1 score* untuk data latih sebesar 100% pada *learning cycle* ke 9, 10, 11, dan seterusnya, sedangkan nilai data uji tertinggi untuk akurasi sebesar 94% pada *learning cycle* ke 1,2,4,6,7 dan 12. Presisi 0,94 pada learning cycle ke 14, *recall* 1 pada *learning cycle* ke 1,2,4,6,7 dan 12
Dan *F1 score* 1 pada *learning cycle* ke 1,2,4,6,7, dan 12. Dari kombinasi data latih dan data uji, diperoleh nilai terbaik pada *learning cycle* ke 12.

c. Pengujian Kinerja Sistem dengan kombinasi jumlah data latih normal dan data uji reproduksi

Tabel 6. Hasil Pengujian Kombinasi Data Latih Normal Dan Data Uji Reproduksi

Learning Cycle	Nilai Akurasi		Nilai Precison		Nilai Recall		Nilai F1 Score	
Load Model	Data Latih (20)	Data Uji (204)	Data Latih (20)	Data Uji (204)	Data Latih (20)	Data Uji (204)	Data Latih (20)	Data Uji (204)
1	80.00%	94.12%	0.76	0.94	1.00	1.00	0.87	0.97
2	70.00%	29.41%	0.89	1.00	0.62	0.25	0.73	0.40
3	90.00%	94.12%	0.87	0.94	1.00	1.00	0.93	0.97
4	85.00%	71.08%	0.81	0.95	1.00	0.73	0.90	0.83
5	100%	92.16%	1.00	0.94	1.00	0.98	1.00	0.96
6	100%	92.16%	1.00	0.94	1.00	0.98	1.00	0.96
7	100%	71.08%	1.00	0.95	1.00	0.73	1.00	0.83
8	100%	92.16%	1.00	0.94	1.00	0.98	1.00	0.96
9	100%	59.31%	1.00	0.94	1.00	0.60	1.00	0.74
10	100%	73.04%	1.00	0.95	1.00	0.75	1.00	0.84
11	100%	61.27%	1.00	0.94	1.00	0.63	1.00	0.75
12	100%	61.27%	1.00	0.94	1.00	0.63	1.00	0.75
13	100%	59.31%	1.00	0.94	1.00	0.60	1.00	0.74
14	100%	61.27%	1.00	0.94	1.00	0.63	1.00	0.75
15	100%	59.31%	1.00	0.94	1.00	0.60	1.00	0.74

Dari tabel 6, diperoleh nilai tertinggi baik untuk akurasi,presisi,*recall*,dan *F1 score* untuk data latih sebesar 100% pada *learning cycle* ke 5, 6, 7, dan seterusnya, sedangkan nilai data uji tertinggi untuk akurasi sebesar 94% pada *learning cycle* ke 1 dan 3, nilai presisi tertinggi sebesar 0.95 pada *learning cycle* ke 4, 7,dan 10. Nilai tertinggi *Recall* sebesar 1 pada *learning cycle* 1 dan 3, dan nilai *F1 score* tertinggi pada *learning cycle* 1 dan 3.

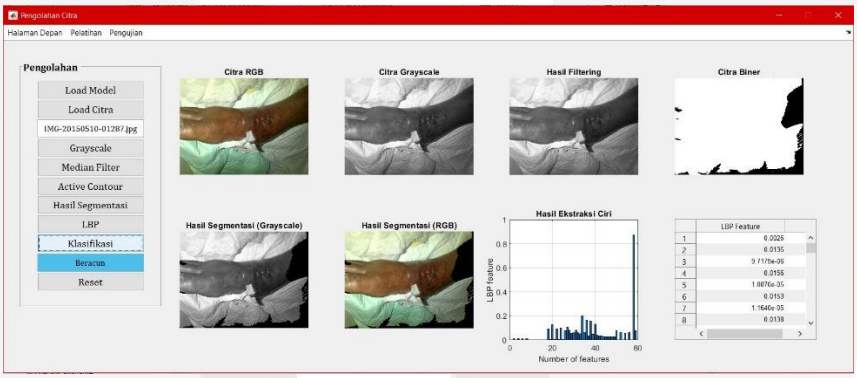
d. Pengujian Kinerja Sistem dengan kombinasi jumlah data latih reproduksi dan data uji reproduksi

Tabel 7. Hasil Pengujian Kombinasi Data Latih Reproduksi Dan Data Uji Reproduksi

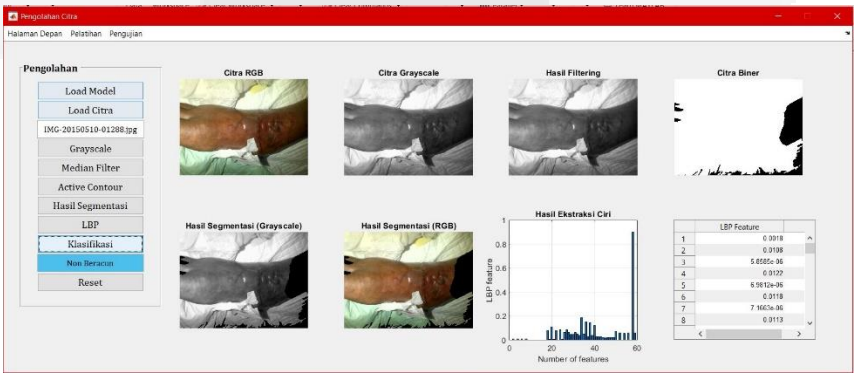
Learning Cycle	Nilai Akurasi		Nilai Precison		Nilai Recall		Nilai F1 Score	
Load Model	Data Latih (80)	Data Uji (204)	Data Latih (80)	Data Uji (204)	Data Latih (80)	Data Uji (204)	Data Latih (80)	Data Uji (204)
1	80.00%	94.12%	0.76	0.94	1.00	1.00	0.87	0.97
2	80.00%	94.12%	0.76	0.94	1.00	1.00	0.87	0.97
3	85.00%	84.31%	0.81	0.93	1.00	0.90	0.90	0.91
4	80.00%	94.12%	0.76	0.94	1.00	1.00	0.87	0.97
5	93.75%	90.20%	0.96	0.94	0.94	0.96	0.95	0.95
6	91.25%	94.12%	0.89	0.94	0.98	1.00	0.94	0.97
7	92.50%	94.12%	0.90	0.94	1.00	1.00	0.95	0.97
8	93.75%	63.24%	1.00	0.98	0.90	0.63	0.95	0.76
9	100.00%	67.16%	1.00	0.98	1.00	0.67	1.00	0.79
10	100.00%	67.16%	1.00	0.98	1.00	0.67	1.00	0.79
11	100.00%	67.16%	1.00	0.98	1.00	0.67	1.00	0.79
12	100.00%	94.12%	1.00	0.94	1.00	1.00	1.00	0.97
13	100.00%	94.12%	1.00	0.98	1.00	0.96	1.00	0.97
14	100.00%	91.67%	1.00	0.99	1.00	0.92	1.00	0.95
15	100.00%	94.12%	1.00	0.98	1.00	0.96	1.00	0.97

Dari tabel 7, diperoleh nilai tertinggi baik untuk akurasi,presisi,*recall*,dan *F1 score* untuk data latih sebesar 100% pada *learning cycle* ke 9, 10, 11, dan seterusnya. Sedangkan nilai data uji tertinggi untuk akurasi sebesar 94%, pada *learning cycle* ke 1,2,4,6,7,12,13,dan 15. Presisi 0,99 pada *learning cycle* ke 14, *recall* 1 pada *learning cycle* ke 1,2,4,6,7,dan 12,*F1 score* 0,97 pada 1,2,4,6,7,12,13,dan 15

Untuk pengolahan citra, didapatkan hasil sebagai berikut:



Gambar 6. Hasil Ekstraksi Fitur IMG-20150510-01287.jpg



Gambar 7. Hasil Ekstraksi Fitur IMG-20150510-01288.jpg

4.3. Analisis Hasil Pengujian

Dari hasil pengujian kinerja sistem untuk variabel *pixel* yang berbeda dengan tujuan mencari *pixel* optimal pada data latih untuk digunakan pada saat pengujian data uji, yaitu 300x300, 400x400, dan 500x500 ternyata didapatkan nilai tertinggi baik dari akurasi, presisi, *recall* dan *F1 Score* berada pada pixel 400x400. Nilai ini didapat dari hasil mengkombinasikan data latih yang sempurna, yaitu nilai akurasi, presisi, *recall* dan *F1 Score* 100% dengan data uji yang ada. Nilai data uji dengan kondisi tersebut adalah akurasinya 94%, presisi 0,94, *recall* 1, dan *F1 score* 0,97, sedangkan untuk *pixel* 300x300 tingkat akurasi hanya 72%, presisi 1, *recall* 0,77 dan *F1 score* 0,85. Untuk *pixel* 500 x 500 nilai akurasinya 94%, presisi 0,94, *recall* 1 dan *F1 score* 0,97, tetapi nilai data uji tersebut didapatkan pada kondisi data latih yang belum sempurna. Hal ini disebabkan pada *pixel* 400x400, sistem bisa menganalisis *detail* citra dengan lebih baik sehingga ekstraksi fitur antar kelas memiliki perbedaan yang signifikan dibandingkan dengan ukuran citra lainnya. Dengan demikian tidak selalu *pixel* yang berukuran lebih besar pasti menghasilkan tingkat akurasi yang lebih tinggi.

Dari tabel 4, 5, 6, dan 7 serta ternyata penambahan jumlah data hasil reproduksi baik data latih maupun data uji dan kombinasi keduanya tidak cukup signifikan dalam mempengaruhi tingkat akurasi, *recall*, dan *F1 score*. Adapun untuk nilai presisi ada kenaikan cukup signifikan jika jumlah data latih lebih tinggi (direproduksi) dari 0,94 menjadi 0,99. Hal ini disebabkan karena adanya peningkatan jumlah data.

Dari gambar 6 (IMG-20150510-01287.jpg) dan 7 (IMG-20150510-01288.jpg), secara umum terlihat kedua citra memiliki kemiripan objek. Perbedaannya gambar 7 merupakan pembesaran dari gambar 6 dengan sudut yang berbeda. Namun, setelah dilakukan pengolahan citra mulai dari *Grayscale*, *Filtering*, Binerisasi, Segmentasi, LBP, dan Klasifikasi dengan *AdaBoost* menghasilkan hasil keputusan yang berbeda. Hal ini kemungkinan disebabkan pada saat akuisisi citra, pembesaran objek dengan sudut yang berbeda,resolusi dan pencahayaan juga berbeda, maka pada saat proses klasifikasi terjadi kesalahan dalam menghasilkan keputusan, sehingga yang mestinya beracun menjadi tidak beracun.

Dari jumlah data uji sebanyak 51 yang terdiri dari 48 citra gigitan ular beracun dan 3 citra gigitan ular tidak beracun sangat tidak proposional. Untuk itu dicoba melakukan reproduksi baik data latih maupun data uji supaya proposional.

5. Kesimpulan

Berdasarkan hasil pengolahan citra dan pengujian kinerja sistem dengan berbagai variabel dalam hal ini *pixel* yaitu 300x300, 400x400 dan 500x500 dan jumlah data, yaitu data latih 20 citra data uji 51 citra serta reproduksi data latih 80 citra dan reproduksi data uji 204 dapat disimpulkan bahwa penggunaan metode LBP dan klasifikasi *AdaBoost* dapat mendeteksi bekas gigitan ular berbisa dan tidak berbisa dengan tingkat akurasi cukup tinggi, yaitu 94%. Disamping itu, sistem menghasilkan nilai presisi, *recall* dan *F1 score* dengan performa mendekati nilai 1. Nilai *pixel* yang optimal yaitu 400x400, karena pada nilai *pixel* tersebut sistem bisa menganalisis *detail* citra dengan lebih baik sehingga ekstraksi fitur antar kelas memiliki perbedaan yang signifikan dibandingkan dengan ukuran citra lainnya.

Penambahan jumlah data hasil reproduksi baik untuk data latih maupun data uji yang tujuan awalnya untuk meningkatkan tingat akurasi ternyata tidak cukup signifikan mempengaruhi tingkat akurasi, yaitu nilai akurasi cenderung stagnan diangka 94%. Hanya untuk nilai presisi mengalami peningkatan pada kombinasi data latih dan data uji reproduksi, yaitu dari awal 0,94 menjadi 0,99. Nilai *recall* dan *F1 score* nya juga tidak berpengaruh signifikan terhadap kombinasi reproduksi data latih maupun data uji.

Kelemahan dari sistem klasifikasi *AdaBoost* terletak pada proses akuisisi citra dan klasifikasi masih dilakukan secara terpisah. Untuk pengembangan selanjutnya, proses akuisisi citra, resolusi dan pencahayaan sebaiknya diseragamkan dengan demikian diharapkan nilai akurasi akan meningkat. Metode yang digunakan juga bisa menggunakan algoritma Jaringan Syaraf Tiruan (JST) atau *Deep Learning*.

Dalam pengolahan citra, adanya pembesaran objek dengan sudut berbeda dapat memepengaruhi hasil keputusan klasifikasi.

Reference

[1] WHO. Snakebite envenoming. 20 Februari 2018. http://www.who.int/news-room/factsheets/detail/snakebite-envenoming. [Diakses 26 Januari 2020].

[2] R. Adiwinata and E. J. Nelwan, "Snakebite in Indonesia," Acta Medica Indones. - Indones. J. Intern. Med., vol. 47, pp. 358–365, 2015.

[3] S. D. A. Nishioka, P. V. P. Silveira, and F. A. Bauab. Bite marks are useful for the differential diagnosis of snakebite in Brazil. Wilderness and Environmental Medicine, 1995.

[4] B. S. Gold, R. C. Dart, and R. A. Barish. Bites of venomous snakes. New England Journal of Medicine, 347(5):347–356, 2002.

[5] R. Fathur, Adiwijaya, and Q. U. Dody. 'Klasifikasi Gigitan Ular Menggunakan Local Binary Pattern dan Naive Bayes' Jurnal Universitas Telkom, 2019.

[6] I. Zaki, and A. T. Hilmy. 'Deteksi dan Klasifikasi Daun Menggunakan Metode Adaboost dan SVM' 2015. https://ojs.amikom.ac.id/index.php/semnasteknomedia/article/view/976. [Diakses 25 Januari 2020]

[7] J. Hubbard. The Survival Doctor's Complete Handbook. Reader's Digest, 2016.

[8] U. D. Dhiya, Adiwijaya, and Q. U. Dody. 'Identifikasi Citra berdasarkan Gigitan Ular menggunakan Active Contour Model dan Support Vector Machine' Jurnal Universitas Telkom, 2019.

[9] Meena, "Local Binary Patterns and Its Variants for Face Recognition," IEEE-International Conf. Recent Trends Inf. Technol., pp. 782–786, 2011.

[10] M. P. Rayiemas, Adiwijaya, and Q. U. Dody. 'Klasifikasi Gigitan Ular Menggunakan Chain Code dan K Nearest Neighbour' Jurnal Universitas Telkom, 2019.

[11] Wing Teng Ho, "Two-stage License Plate Detection Using Gentle Adaboost," First Asian Conference on Intellegent Information and Database System, pp. 109-114, 2009.

[12] F.Yoav and E.S.Robert "A Short Introduction to Boosting," Journal of Japanese Society for Artificial Intelligence, 14(5), pp. 771-780, 1999.

[13] Z. Zhi-Hua. 2012, Ensemble Methods: Foundations and Algorithms, Boca Raton, Florida: Chapman & Hall/CRC.

[14] Viola, Paul A. and Jones, Michael J. "Rapid Object Detection using a Boosted Cascade of Simple Features", IEEE CVPR, 2001.

[15] Esa Prakasa," Texture Feature Extraction by Using Local Binary Pattern". INKOM, Vol. X, No. x, 2015.

[16] T. Ahonen and M. Pietik¨ainen, "A framework for analyzing texture descriptors," Threshold, vol. 5, no 9, p. 1, 2008.

[17] R.Septia, S.Deni, and Jondri. 'Klasifikasi Jenis Kelamin Berdasarkan Citra Wajah Menggunakan Kombinasi Algoritma Adaboost – Support Vector Machine' Jurnal Universitas Telkom, 2011.