BAB 2
LANDASAN TEORI

2.1 Tinjauan Teori

Dalam klasifikasi motif batik Trusmi, terdapat beberapa teori yang dipahami seperti algoritma CNN, *Saliency Map*, dan teori-teori lainnya.

2.1.1 Batik Trusmi

Batik adalah teknik pewarnaan kain tradisional yang menggunakan malam atau lilin sebagai perintang warna, membentuk pola atau motif khas pada kain. Seni batik di Indonesia telah ada sejak zaman kerajaan Jawa kuno dan mencerminkan status sosial serta budaya masyarakat. Proses pembuatan batik meliputi pencantingan motif dengan canting atau cap, serta pewarnaan berulang untuk mencapai hasil yang diinginkan. Batik di setiap daerah memiliki kekhasan motif, seperti motif Mega Mendung dari Cirebon dan motif Kawung serta Parang dari Jawa Tengah, yang masing-masing memiliki makna simbolis. Salah satu pusat batik di Cirebon adalah Desa Trusmi, yang berkembang sejak era Kerajaan Cirebon pada abad ke-14. Berawal dari peran Ki Buyut Trusmi bersama Sunan Gunung Jati yang menyebarkan Islam dan mengajarkan keterampilan membatik, desa ini menjadi sentra pengrajin batik yang diakui keahliannya oleh Sultan Keraton Cirebon. Hingga kini, batik Trusmi terus berkembang dan menjadi bagian penting dari identitas budaya Cirebon. Pada tahun 2009, UNESCO mengakui batik sebagai Warisan Budaya Takbenda, menggarisbawahi nilai-nilai budaya yang terkandung dalam teknik ini. [23, 5]

2.1.2 Convolutional Neuron Network

Convolutional Neural Network (CNN) adalah jenis jaringan saraf tiruan yang dirancang untuk memproses data *grid-like*, seperti gambar. CNN diklaim sebagai model terbaik untuk memecahkan masalah dalam pengenalan objek [24, 25].CNN dikembangkan berdasarkan *Multilayer Perceptron* (MLP) yang dirancang untuk memproses data dua dimensi.

CNN pertama kali dikembangkan oleh Kunihiki Fukushima dengan nama *NeoCognitron* yang kemudian dikembangkan oleh Yann LeChun untuk

pengenalan angka dan tulisan tangan. Dengan kemenangan Alex Krizhevsky pada kompetisi *ImageNet Large Scale Visual Recognition Challenge* di tahun 2012 dengan menggunakan aplikasi CNN, membuktikan bahwa metode *Deep Learning* menggunakan CNN lebih mumpuni dibandingkan metode lainnya seperti SVM dalam klasifikasi objek pada gambar.

CNN terdiri dari beberapa jenis lapisan utama [1, 26]:

1. Convolutional Layer

   Lapisan ini berfungsi untuk mengekstrak fitur dari input data. Operasi konvolusi ini melibatkan dua fungsi dengan argumen berupa nilai nyata. Melalui operasi ini, input citra diolah untuk menghasilkan *Feature Map* (peta fitur) sebagai fungsi output. Baik input maupun output dalam proses ini terdiri dari argumen dengan nilai nyata. Operasi konvolusi secara umum dapat dituliskan dengan rumus di bawah ini:

   $$s(t) = (x * \omega) \tag{2.1}$$

   Persamaan 2.1 menghasilkan peta fitur sebagai output tunggal dengan dua argumen. Input yang diwakili oleh x dan kernel atau fitur yang diwakili oleh ω. Karena input berupa gambar dua dimensi, maka t dapat diganti dengan argumen i dan j. Oleh karena itu, operasi konvolusi dengan input lebih dari satu dimensi dituliskan sebagai berikut:

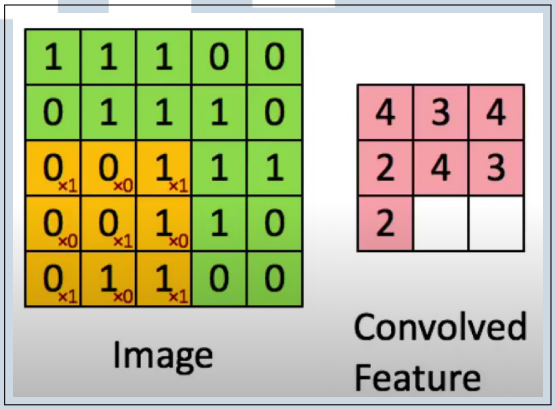
   $$s(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n) \tag{2.2}$$

   Persamaan 2.2 merupakan perhitungan dasar untuk operasi konvolusi, dimana piksel gambar yang dinyatakan sebagai i dan j. Persamaan 2.2 bersifat komutatif ketika K sebagai kernel dapat diganti dengan I sebagai input. Volume keluaran dapat diatur menggunakan hyperparameter. Hyperparameter digunakan untuk menghitung jumlah neuron aktivasi dalam satu output yang dinyatakan dalam persamaan berikut:

   $$\frac{(W - F + 2P)}{S + 1} \tag{2.3}$$

   Dari persamaan 2.3, ukuran spasial dari output volume dapat dihitung dengan hyperparameter yang digunakan adalah ukuran volume (W), filter (F), *Stride* yang diterapkan (S), dan jumlah *zero padding* yang digunakan (P). *Stride*

adalah nilai yang digunakan untuk menggeser filter melalui input gambar dan *Zero Padding* adalah nilai untuk menempatkan angka nol di sekitar batas gambar. Dalam pemrosesan gambar, konvolusi berarti menerapkan kernel (kotak kuning) ke gambar di semua kemungkinan offset seperti yang ditunjukkan pada gambar 2.1.



Gambar 2.1. Convolutional Operation.
Sumber: [1].

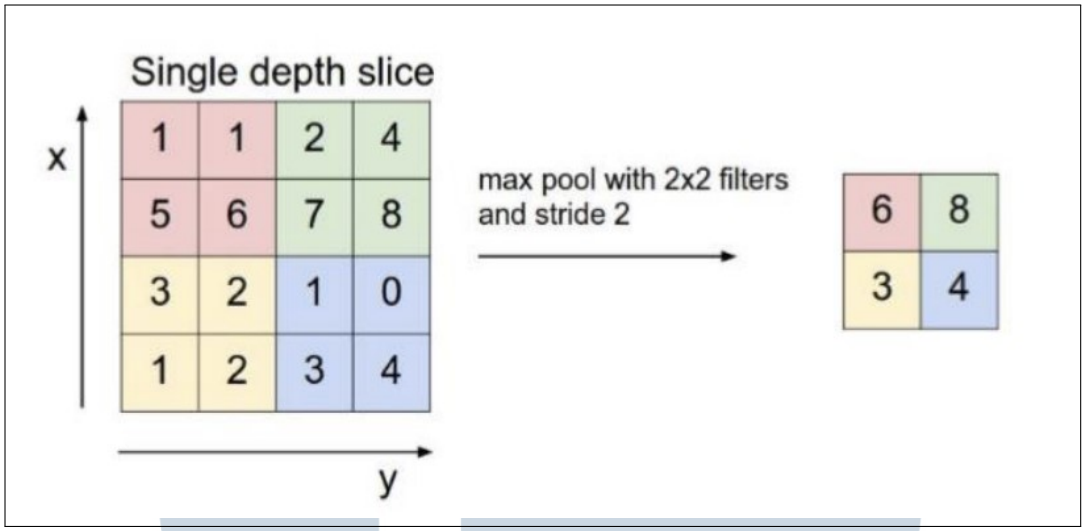
Kotak hijau secara keseluruhan adalah gambar yang akan konvolusi. Kernel bergerak dari sudut kiri atas ke kanan bawah. Sehingga konvolusi dari gambar tersebut dapat dapat dilihat pada gambar di sebelah kanan. Tujuan dari konvolusi pada data citra adalah untuk mengekstrak fitur dari citra masukan.

2. Pooling Layer

   Lapisan ini berfungsi untuk mengurangi dimensionalitas dari fitur yang diekstrak, sehingga mengurangi kompleksitas komputasi dan mengontrol *overfitting*. Terdapat beberapa jenis *pooling layer* yaitu, *max pooling*, *average pooling*, dan *soft pooling*. Umumnya, digunakan *max pooling* sebagai metode penggabungan yang digunakan dalam CNN.

   (a) Max Pooling

   Max pooling membagi output dari konvolusi layer menjadi beberapa *grid* kecil dan kemudian mengambil nilai maksimum dari setiap *grid* [27]. *Grid* digunakan untuk menyusun matriks gambar yang telah direduksi seperti yang seperti yang ditunjukkan pada gambar 2.2.



Gambar 2.2. Max Pooling Operation
Sumber: [1]

Grid yang berwarna merah, hijau, kuning dan biru adalah kelompok *grid* yang akan dipilih nilai maksimumnya. Sehingga hasil dari proses tersebut dapat dilihat pada kumpulan *grid* di sebelah sebelah kanan. Proses tersebut memastikan bahwa fitur yang didapatkan akan sama meskipun objek gambar mengalami pergeseran (translasi). Bentuk pooling ini akan mengurangi *Feature Map* hingga 75% dari ukuran aslinya. *Max pooling* dihitung menggunakan persamaan 2.4

$$f_{\max}(\mathbf{x}) = \max\{x_i\}_{i=1}^{N} \tag{2.4}$$

(b) Average Pooling

Average pooling merangkum seluruh fitur yang terdapat dalam bagian *pooling*. *Average pooling* dihitung menggunakan persamaan 2.5

$$f_{\text{avg}}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N}|x_i| \tag{2.5}$$

Penggunaan *average pooling* dapat mengurangi pengaruh dari fitur-fitur yang kurang relevan. Tetapi, karena diberikan bobot kepentingan yang sama maka dapat menyebabkan *background region* lebih mendominasi dalam representasi gabungan [27].

(c) Soft Pooling

Soft pooling digunakan sebagai cara perantara antara *max pooling*

dan *average pooling*. *Soft pooling* menggunakan fungsi-fungsi yang dapat dibedakan secara halus untuk memperkirakan maks dan rata-rata penggabungan untuk pengaturan parameter yang berbeda. *Soft pooling* dihitung menggunakan persamaan 2.6

$$f_{\mathrm{GM}} = \left(\frac{1}{N} \sum_{i=1}^{N} |x_i|^r \right)^{\frac{1}{r}} \tag{2.6}$$

Parameter r menentukan tingkat kelembutan, apabila r bernilai 1 maka fungsi setara dengan *average pooling*, jika r bernilai tak hingga maka fungsi setara dengan *max pooling* [27].

3. Augmentasi

   Augmentasi dilakukan untuk meningkatkan jumlah data secara virtual berdasarkan data yang sudah disediakan sehingga sistem lebih kuat dan generalis terhadap variasi yang ada. Pada penelitian kali ini, digunakan fungsi imageDataGenerator berdasarkan *library* Keras. Dalam pengenalan gambar, teknik augmentasi data yang umum meliputi *flipping* (membalik gambar secara horizontal atau vertikal), *rotation* (memutar gambar pada sudut tertentu), *cropping* (memotong bagian tertentu dari gambar), dan *color jittering* (mengubah kecerahan, kontras, dan saturasi gambar).

(a) Hasil Augmentasi Horizontal Flip 1 (b) Hasil Augmentasi Horizontal Flip 2

Gambar 2.3. Hasil Augmentasi Horizontal Flip
Sumber: [2]

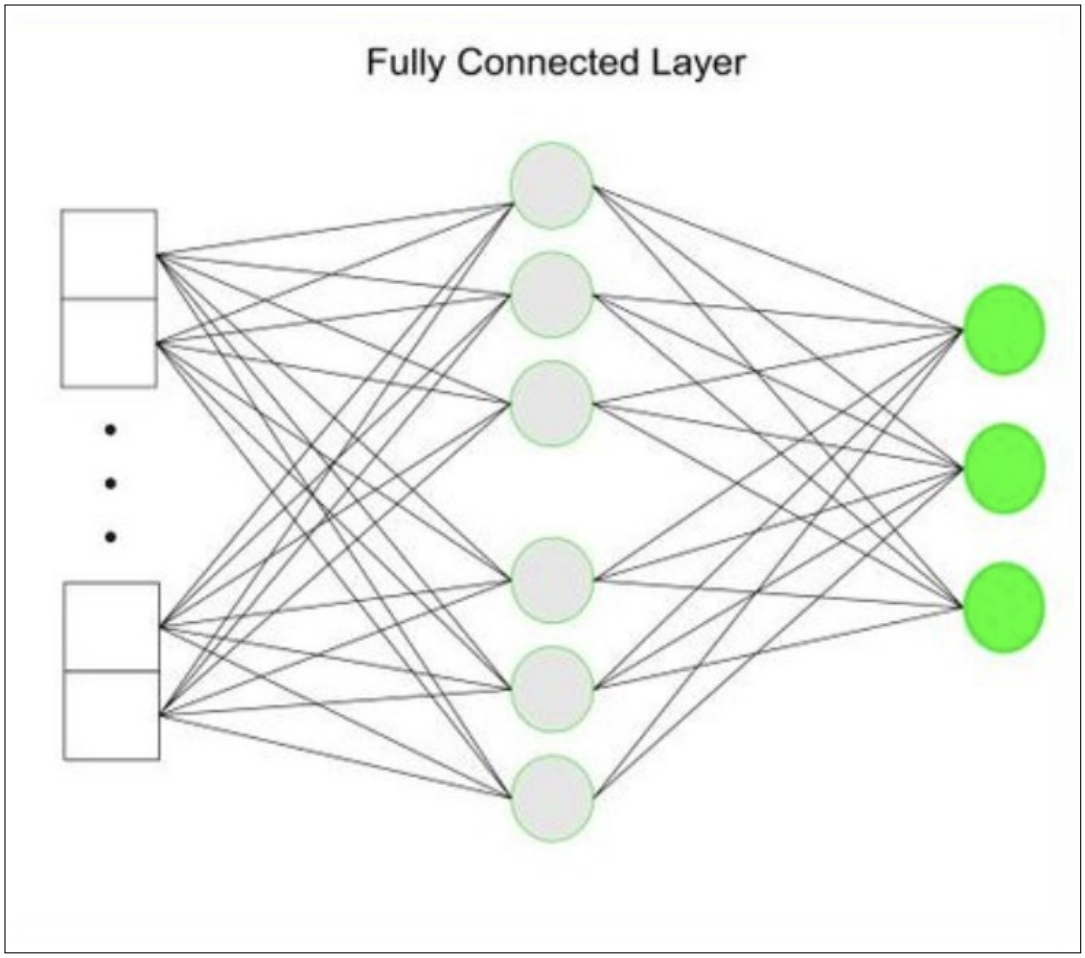
Gambar 2.3 menunjukkan contoh hasil augmentasi data *flipping*, dimana gambar 2.3a dibalik secara horizontal sehingga menghasilkan gambar 2.3b.

(a) Hasil Augmentasi Vertikal Flip 1 (b) Hasil Augmentasi Vertikal Flip 2

Gambar 2.4. Hasil Augmentasi Vertikal Flip
Sumber: [2]

Gambar 2.4 menunjukkan contoh hasil augmentasi data *flipping*, dimana gambar 2.4a dibalik secara vertikal sehingga menghasilkan gambar 2.4b.

4. Fully Connected Layer
   Setelah beberapa lapisan konvolusi dan *pooling*, lapisan ini menghubungkan setiap neuron ke semua neuron di lapisan sebelumnya, yang umumnya digunakan untuk klasifikasi. Langkah ini digunakan pada akhir jaringan karena setiap neuron perlu diubah menjadi data satu dimensi sehingga tidak dapat diubah kembali menjadi data awal.



Gambar 2.5. Fully connected layer
Sumber: [3]

Gambar 2.5 menunjukkan ilustrasi dari *Fully Connected Layer*. Kotak di sebelah kiri merepresentasikan input ke jaringan, dimana setiap kotak mewakili 1 fitur dari data input. Lingkaran abu-abu di tengah adalah neuron pada lapisan tersembunyi dari jaringan saraf. Setiap neuron menerima informasi dari semua neuron di lapisan sebelumnya *input layer*. Neuron pada lapisan tersembunyi ini memproses informasi dengan menerapkan bobot dan bias pada input, kemudian mengirim hasilnya melalui fungsi aktivasi. Lingkaran hijau di sebelah kanan merepresentasikan neuron output. Output dari neuron ini akan menjadi hasil akhir dari jaringan saraf. Setiap neuron output menghasilkan hasil yang bergantung pada nilai-nilai dari neuron di lapisan tersembunyi. Sedangkan garis-garis yang menghubungkan neuron-neuron adalah koneksi antara neuron, yang disebut bobot. Setiap garis menunjukkan bahwa neuron di lapisan sebelumnya terhubung ke neuron di

lapisan berikutnya. Bobot pada garis-garis ini berfungsi untuk menyesuaikan pengaruh dari setiap input.

Pada lapisan tersembunyi, setiap neuron menerima input yang telah dikalikan dengan bobot dan ditambahkan bias. Nilai input ke neuron dihitung menggunakan persamaan 2.7.

$$z = (w_1 \times x_1) + (w_2 \times x_2) + \cdots + (w_n \times x_n) + b \tag{2.7}$$

di mana w adalah bobot, x adalah nilai input, dan b adalah bias. Setelah nilai z dihitung, fungsi aktivasi diterapkan untuk memperkenalkan non-linearitas dalam model, yang penting untuk memungkinkan jaringan saraf belajar pola yang kompleks dalam data.

2.1.3 Activation Function

Terdapat beberapa jenis fungsi aktivasi yaitu, *Rectified Linear Units* (ReLu), *Sigmoid*, *Hyperbolic Tangent* (Tanh), *Thresholded ReLu* (T-ReLu), dan *Softmax Classifier*.

Aktivasi *Sigmoid* merupakan fungsi yang memetakan input ke rentang (0, 1) dan sering digunakan dalam layer output untuk masalah klasifikasi biner. Aktivasi *Sigmoid* dihitung menggunakan persamaan 2.8.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.8}$$

Aktivasi Tanh merupakan fungsi yang memetakan input ke rentang (-1, 1) dan sering digunakan di lapisan tersembunyi. Aktivasi Tanh dihitung menggunakan persamaan 2.9.

$$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{2.9}$$

Aktivasi T-ReLu merupakan fungsi Varian dari ReLU yang memperkenalkan ambang batas (*threshold*). Nilai input yang lebih kecil dari *threshold* akan menjadi nol. Aktivasi T-ReLu dihitung menggunakan persamaan 2.10.

$$f(x) = \begin{cases} x & \text{jika } x > \theta \\ 0 & \text{jika } x \leq \theta \end{cases} \tag{2.10}$$

Fungsi aktivasi yang akan digunakan adalah *Rectified Linear Units* (ReLu)

dan *Softmax Classifier*. Aktivasi ReLu meningkatkan sifat non-linear dari fungsi pengambilan keputusan dan semua jaringan tanpa mempengaruhi bidang reseptif dari *Convolutional Layer*. ReLu juga digunakan secara luas karena dapat melatih jaringan syaraf lebih cepat. Aktivasi ReLu dihitung menggunakan persamaan 2.11.

$$f(x) = \max(0, x) \tag{2.11}$$

Aktivasi *Softmax* untuk lapisan ini adalah bentuk lain dari algoritma Regresi Logistik yang dapat digunakan untuk mengklasifikasikan lebih dari dua kelas. *Softmax* memberikan hasil yang lebih intuitif dan memiliki interpretasi probabilistik yang lebih baik daripada algoritma klasifikasi algoritma lainnya. *Softmax* memungkinkan untuk menghitung probabilitas untuk semua label. Dari label yang ada, sebuah nilai nyata diambil dan mengubahnya menjadi vektor dengan nilai antara nol dan satu yang jumlahnya akan bernilai satu [1]. Aktivasi *Softmax* dihitung menggunakan persamaan 2.12.

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{2.12}$$

dengan z adalah faktor input, z_i adalah komponen dari vektor, dan K adalah jumlah kelas.

2.1.4 Saliency Map

Saliency map adalah konsep penting dalam pengenalan pola dan visi komputer yang digunakan untuk mengidentifikasi area paling signifikan dalam sebuah gambar yang mempengaruhi keputusan model [28]. Teori dasar *Saliency Map* berakar pada bagaimana sistem visual manusia bekerja, di mana otak kita secara otomatis fokus pada bagian-bagian yang menonjol atau menarik perhatian dalam suatu pemandangan. Dalam konteks pembelajaran mesin dan jaringan saraf tiruan, *Saliency Map* berfungsi untuk memberikan interpretasi visual dari proses pengambilan keputusan model, dengan menyoroti piksel-piksel dalam gambar yang paling mempengaruhi hasil klasifikasi atau deteksi. Beberapa metode utama dalam pembuatan *Saliency map* yaitu, gradien, *Class Activation Maps* (CAM), dan Grad-CAM.

Pada penelitian ini digunakan metode gradien yang berfokus untuk menghitung sensitivitas prediksi model terhadap input gambar. Dalam metode ini,

gradien dari skor kelas terhadap gambar input dihitung untuk menentukan piksel mana yang paling mempengaruhi prediksi model. Fitur-fitur *Saliency map* yang digunakan sebagai berikut:

1. Gradien
   Gradien dari skor kelas terhadap gambar input menunjukkan sensitivitas atau pengaruh dari setiap piksel pada prediksi model. Piksel dengan gradien besar berarti perubahan kecil pada piksel tersebut akan sangat mempengaruhi hasil prediksi [4].

2. Reduksi saluran (Channel-wise Reduction)
   Menjumlahkan gradien sepanjang channel warna menghasilkan peta saliency yang lebih sederhana dan mudah diinterpretasikan, mengabaikan informasi warna yang spesifik untuk memberikan gambaran umum tentang pentingnya area dalam gambar [29].

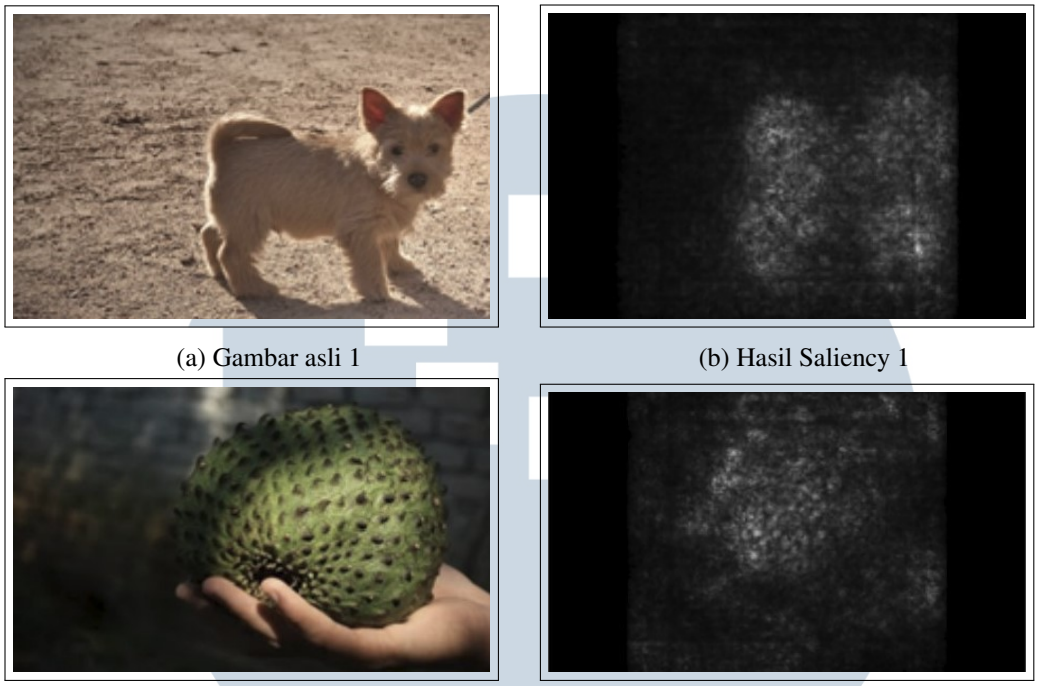
3. Normalisasi
   Normalisasi membantu dalam visualisasi dengan mengatur rentang nilai piksel ke dalam batas yang dapat dilihat (0-255 untuk gambar grayscale), sehingga memudahkan identifikasi area penting [30].

4. Overlay dan visualisasi warna
   Menggunakan *colormap* dan *overlay saliency map* pada gambar asli membantu dalam memahami secara visual area mana yang paling berkontribusi terhadap keputusan model, memberikan wawasan yang lebih baik [31].

Saliency map dihitung menggunakan persamaan 2.13.

$$\mathbf{M} = \left| \frac{\partial S_c}{\partial \mathbf{I}} \right| \tag{2.13}$$

dengan M merupakan *saliency map*, S_c merupakan nilai untuk kelas c, I merupakan input gambar, dan $\left| \frac{\partial S_c}{\partial \mathbf{I}} \right|$ merupakan gradien skor sesuai dengan input gambar [4].



(a) Gambar asli 1 (b) Hasil Saliency 1

(c) Gambar asli 2 (d) Hasil Saliency 2

Gambar 2.6. Saliency Map
Sumber: [4]

Gambar 2.6 menunjukkan contoh keluaran dari *saliency map*. Gambar 2.6b dan gambar 2.6d menunjukkan area yang paling signifikan dalam mempengaruhi keputusan model berdasarkan gambar 2.6a dan gambar 2.6c.

2.1.5 Evaluasi

Evaluasi dilakukan untuk menentukan tingkat keberhasilan dari model yang sudah dibangun. Model akan dites dengan membandingkan antara data latih dengan data testing berdasarkan beberapa parameter yaitu, akurasi, presisi, *recall*, dan nilai F1 yang dihitung berdasarkan *confusion matrix*.

Confusion matrix digunakan secara luas dalam pembelajaran mesin untuk klasifikasi terawasi atau untuk memahami perilaku model klasifikasi. Matriks ini memiliki struktur persegi yang direpresentasikan dengan baris dan kolom, di mana baris menunjukkan kelas sebenarnya dari data dan kolom menunjukkan kelas yang diprediksi. Dalam klasifikasi biner, *confusion matrix* berbentuk matriks 2 x 2. Matriks ini mencakup empat ukuran utama: *true positive* (TP), *true negative* (TN), *false positive* (FP), dan *false negative* (FN). Untuk masalah klasifikasi multikelas, *confusion matrix* dengan k kelas akan memiliki matriks kebingungan k x k [32].

Tabel 2.1 menunjukkan contoh bentuk *confusion matrix* 2 x 2.

Tabel 2.1. Confusion Matrix

		Predict	
		Positive	Negative
Actual	Positive	True positive (TP)	False negative (FN)
	Negative	False positive (FP)	True Negative (TN)

Nilai presisi (P) adalah rasio observasi positif yang diprediksi secara akurat terhadap total observasi positif yang diprediksi. Nilai presisi dihitung menggunakan persamaan 2.14 [33].

$$P = \frac{TP}{TP+FP} \times 100\% \tag{2.14}$$

Nilai *recall* (R) adalah rasio pengamatan positif yang diprediksi secara akurat terhadap jumlah semua sampel yang relevan. Nilai *recall* dihitung menggunakan persamaan 2.15 [33].

$$R = \frac{TP}{TP+FN} \times 100\% \tag{2.15}$$

Skor F1 adalah rata-rata tertimbang dari presisi dan *recall* dan metode yang digunakan untuk mengukur kinerja model. Skor F1 dihitung menggunakan persamaan 2.16 [33].

$$F_1 \text{ score} = \frac{2PR}{P+R} \tag{2.16}$$

Nilai akurasi adalah persentase dari data yang diklasifikasikan dengan benar. Nilai akurasi merupakan ukuran kinerja yang paling intuitif dan merupakan rasio dari observasi yang diprediksi dengan benar terhadap total pengamatan. Nilai akurasi dihitung menggunakan persamaan 2.17 [33].

$$\text{Acc} = \frac{TP+TN}{TP+TN+FP+FN} \times 100\% \tag{2.17}$$