

## **Nutify: Identifikasi Gambar Manusia untuk Mencegah Pornografi Menggunakan Metode Hebbian Learning dan Back Propagation Network**

Guntur Prabawa Kusuma<sup>1</sup>, Suyanto<sup>2</sup>, Fazmah Arif Yulianto<sup>3</sup>

<sup>1</sup>Program Studi Manajemen Informatika, Politeknik Telkom Bandung

<sup>2,3</sup>Fakultas Teknik Informatika, Institut Teknologi Bandung

<sup>1</sup>guntur@politekniktelkom.ac.id, <sup>2</sup>suy@ittelkom.ac.id, <sup>3</sup>faz@ittelkom.ac.id

---

### **Abstrak**

Penelitian ini mempresentasikan suatu metode pengenalan objek pada citra digital yang dapat mengklasifikasikan citra tersebut apakah terdapat objek manusia telanjang atau tidak. Tahapan-tahapan identifikasi adalah: pertama, mencari daerah yang terdapat warna kulit; kedua, adalah dengan klasifikasi pola dimana setiap piksel dalam sebuah citra sebagai sebuah koordinat dalam ruang berdimensi tinggi diproyeksikan ke dalam sebuah ruang berdimensi rendah menggunakan teknik PCA (*Principal Component Analysis*) dengan metode Hebbian Learning; ketiga, adalah menggunakan Back Propagation Network sebagai metode klasifikasi. Dari hasil pengujian yang dilakukan pada Nutify, tingkat akurasi pengenalan gambar manusia telanjang menggunakan metode Hebbian Learning dan Back Propagation Network mencapai 75,2459 % untuk pengujian sebanyak 610 citra, dengan waktu pengenalan satu citra mencapai 8 detik.

**Kata kunci:** citra digital, piksel, identifikasi citra manusia, PCA (*principal component analysis*), hebbian learning, back propagation network

---

### **Abstract**

This research represents a digital image's object recognition method that classifies image whether it contains nude picture or not. The identification process are : First, locating inside an image area that contain skin color; Second, pattern classification where each pixel in an image as a coordinate in high dimension space projected into lower dimension space using PCA (Principal Component Analysis) technique with Hebbian Learning method; Third, using Back Propagation Network as the classification method. As tested with 610 images, Nutify system shows the accuracy of this method is 75,25% with recognition time for one image is 8 seconds.

**Keywords:** digital image, pixel, PCA (principal component analysis), hebbian learning, back propagation network

---

### **1. Pendahuluan**

Munculnya tindakan-tindakan asusila sering dipicu dari kebiasaan pelaku mengkonsumsi pornografi baik dalam bentuk VCD, gambar-gambar porno, cerita-cerita porno, dan lain sebagainya.

Situs-situs ini dapat diakses dengan mudah oleh siapapun yang mengerti bagaimana membuka dan mengoperasikan *web browser*, menggunakan *search engine* dan menuju ke portal yang disediakan. Bahkan seorang anak 10 tahun mampu mendapatkan materi pornografi hanya dengan memasukkan beberapa *keyword* tanpa ada batasan. Sekarang ini memang sudah banyak cara yang dapat digunakan untuk membatasi atau menghalangi para *netter* dalam mengakses situs porno seperti pemeriksaan pada alamat situs yang akan diakses. Jika terdapat kata-kata kunci : xxx, *nude*, *porn*, porno, *adult*, atau yang mengandung makna pornografi, maka secara otomatis koneksi ke situs tersebut akan digagalkan.

Tetapi sekarang ini justru banyak sekali alamat situs porno yang tidak mengandung kata-kata kunci

tersebut, sehingga para *netter* tetap saja bisa mengakses pornografi melalui internet.

Oleh karena itu dibutuhkan suatu cara untuk mengenali apakah suatu gambar termasuk dalam kategori porno atau tidak. Dalam penelitian ini, diamati salah satu bagian dari beberapa komponen pornografi, yaitu gambar manusia telanjang. Menurut Hazel (1983), warna kulit manusia disusun oleh warna darah (merah) dan melanin (kuning dan coklat). Salah satu metode yang dapat digunakan untuk mengenali ada/tidaknya objek manusia telanjang dalam suatu citra masukan adalah dengan melakukan proses segmentasi atau pengelompokan area warna kulit [1]. Langkah berikutnya dilakukan proses ekstraksi ciri menggunakan metode Hebbian Learning [3] dari area-area yang didapatkan dan kemudian melakukan klasifikasi keputusan.

Terdapat beberapa aspek untuk merumuskan permasalahan yang ada, yaitu: aspek pertama adalah bagaimana melakukan proses segmentasi antara warna kulit dengan yang bukan kulit pada citra masukan (aspek pertama tidak dibahas lebih dalam

pada penelitian ini). Aspek kedua adalah bagaimana cara melakukan ekstraksi untuk mendapatkan ciri-ciri dari setiap citra masukan. Aspek ketiga adalah bagaimana cara melakukan klasifikasi citra masukan dengan melakukan pengukuran kemiripan ciri-ciri dari citra masukan dengan ciri-ciri yang sudah dilatihkan pada suatu jaringan syaraf tiruan sehingga dapat ditentukan apakah gambar tersebut termasuk dalam kategori telanjang atau tidak.

## 2. Pembahasan

### 2.1 Citra Digital

Citra, atau yang lebih kita kenal sebagai gambar, adalah representasi dua dimensi dari kenyataan yang mampu direspon oleh persepsi visual manusia. Ada banyak macam representasi citra yang kita kenal. Misalnya lukisan, gambar, foto, dan lain-lain.

Citra, atau *image* atau gambar, dapat didefinisikan sebagai fungsi intensitas cahaya dua dimensi  $f(x, y)$ , dengan  $x$  dan  $y$  merupakan koordinat spasial dan nilai  $f$  pada titik berapapun  $(x, y)$  adalah proporsional terhadap *brightness* (level keabuan) dari gambar tersebut pada titik itu. Karena  $f(x, y)$  merupakan fungsi intensitas cahaya, maka  $f$  merupakan bentuk energi sehingga memiliki daerah intensitas dari nol sampai dengan tak hingga:  $0 < f(x, y) < \infty$

Citra digital adalah sebuah representasi citra  $f(x, y)$  dalam bentuk diskrit pada kedua dimensinya, yakni secara koordinat spasial maupun level keabuannya (intensitas). Kita dapat menganggap citra digital sebagai sebuah matriks yang baris dan kolomnya mengidentifikasi *level* keabuan pada titik tersebut [2].

### 2.2 Skin Color Segmentation

Algoritma untuk mengidentifikasi kulit manusia sudah menjadi hal yang umum dalam setiap sistem ‘penglihatan’ pada komputer. Tujuan dari segmentasi kulit adalah untuk menghilangkan daerah-daerah mana saja pada citra yang tidak mengandung warna kulit.

Warna kulit manusia dihasilkan dari kombinasi antara warna darah (merah) dan melanin (kuning dan coklat) [4] serta memiliki rentang *hue* yang terbatas. Warna kulit kadang-kadang tersaturasi, tetapi tidak tersaturasi secara ekstrim karena untuk warna kulit yang dalam, dihasilkan dengan menambahkan melanin. Saturasi akan meningkat jika warna kulit semakin kuning.

a. *Hue*, suatu nilai yang menunjukkan kualitas perbedaan yang jelas antar warna, contoh : merah, hijau, kuning atau ungu[4].

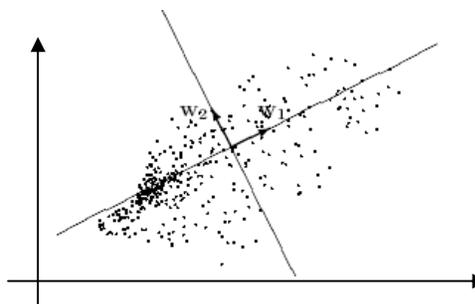
$$Hue = \arctan(R_g, B_y)$$

b. *Saturation*, mengacu pada intensitas keabuan sebuah warna[4].

$$Sat = \sqrt{R_g^2 - B_y^2}$$

### 2.3 Hebbian Learning

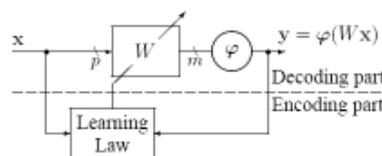
Hebbian Learning termasuk dalam *Unsupervised Learning* (pembelajaran tanpa pengawasan) yang digunakan untuk mengekstraksi data berupa *principal directions* (arah-arah prinsipal) dari kumpulan data yang dapat dipetakan kedalam ruang  $p$ -dimensi. Setiap arah yang digambarkan sebagai bobot vektor dan jumlah *principal direction* biasanya sama dengan jumlah dimensi data dari ruang input  $p$ . Sebagai gambaran, berikut contoh pemetaan data dua dimensi dengan bobot vektor  $w_1$  dan  $w_2$ :



Gambar 1. Pola Pemetaan 2 Dimensi dengan Vektor Prinsipal.

#### 2.3.1 Struktur Dasar Jaringan Syaraf Tiruan Hebbian Learning

Struktur dasar dari jaringan syaraf tiruan (JST) yang menggunakan algoritma Hebbian Learning terdiri dari sebuah *layer decoder* dan modul pembelajaran.



Gambar 2. Diagram Blok dari JST Hebbian Learning

Pada *layer decoder* terdapat sebuah matriks bobot  $W$  yaitu  $m \times p$ . Setiap baris bobot vektor dihubungkan dengan satu *neuron*. *Layer decoder* biasanya linier, sehingga dapat menyederhanakan struktur jaringan. Kompleksitas jaringan bergantung dari struktur aturan pembelajaran yang digunakan.

#### 2.3.2 Aturan Dasar Hebbian Learning

Ide dasar dari proses Hebbian Learning adalah dengan melakukan *update* pada bobot sinaptik yang

sebanding dengan sinyal *input* dan *output*, persamaannya adalah sebagai berikut :

$$\begin{aligned} w_{ji}(n+1) &= f(w_{ji}(n), y_j(n), x_i(n)) \\ w_{ji}(n+1) &= w_{ji}(n) + \eta y_j(n) x_i(n) \end{aligned} \quad (1)$$

dimana  $w_{ji}(n)$  adalah bobot sinaptik waktu ke- $n$ , jadi  $w_{ji}(n)$  dan  $w_{ji}(n+1)$  adalah nilai bobot-bobot sinaptik yang lama dan yang baru.  $\eta$  adalah suatu konstanta positif sebagai *learning rate*.

Persamaan di atas pada dasarnya masih belum stabil, hal ini ditunjukkan oleh perkembangan bobot-bobotnya yang tidak terbatas selama proses pembelajaran

Untuk menstabilkan metode Hebbian *Learning* dibutuhkan 2 langkah dasar, yaitu[3]:

1. Anggap  $\mathbf{x}$  adalah vektor *random* dengan  $p$ -dimensi yang mewakili data input, maka dibutuhkan *mean* (rata-rata) adalah nol:  $E[\mathbf{x}] = 0$ . Jika rata-ratanya nol, maka matriks korelasi akan sama dengan matriks kovarian.
2. Aturan dasar Hebbian *Learning* dimodifikasi sedemikian rupa sehingga *magnitude* dari bobot vektor-vektor cenderung mendekati 1 dan dapat dicapai dengan normalisasi berikut:

$$\mathbf{w} \leftarrow \mathbf{w} + \eta \mathbf{y} \mathbf{x}^T ; \mathbf{w} \leftarrow \mathbf{w} / \|\mathbf{w}\| \quad (2)$$

dimana *magnitude* vektor dihitung dengan cara:

$$\|\mathbf{w}\| = \sqrt{\sum_{i=1}^p w_i^2}, \text{ serta } \|\mathbf{w}\|^2 = \mathbf{w} \mathbf{w}^T \quad (3)$$

Jika menggunakan persamaan (2) untuk melakukan normalisasi, akan mengalami kesulitan dalam hal komputasi, maka dilakukan penurunan dengan perluasan Deret Taylor dan didapatkan persamaan sebagai berikut:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta y(n) (\mathbf{x}^T(n) - y(n) \mathbf{w}(n)) \quad (4)$$

persamaan ini akan menghasilkan *magnitude* dari bobot vektor mendekati 1,  $\|\mathbf{w}\| \rightarrow 1$ . Persamaan (4) inilah yang akan digunakan untuk meng-*update* bobot vektor yang digunakan dalam Hebbian *Learning*.

### 2.3.3 Hebbian Learning untuk Menghitung PCA

#### A. Ekstraksi *Principal Direction*

Algoritma pembelajaran yang digambarkan oleh persamaan (4) untuk kasus *neuron* tunggal, disebut sebagai “aturan Oja”[3], dapat dituliskan ke dalam bentuk berikut:

$$\begin{aligned} \Delta \mathbf{w} &= \eta \mathbf{y} (\mathbf{x}^T - \mathbf{y} \mathbf{w}) = \eta \mathbf{y} \tilde{\mathbf{x}}^T, \quad \text{dimana} \\ \mathbf{y} &= \mathbf{w} \mathbf{x} = \mathbf{x}^T \mathbf{w}^T \end{aligned} \quad (5)$$

dengan vektor perbesaran input adalah:

$$\tilde{\mathbf{x}} = \mathbf{x} - \mathbf{y} \mathbf{w}^T \quad (6)$$

Substitusi persamaan (6) ke dalam persamaan (5) menghasilkan:

$$\Delta \mathbf{w} = \eta (\mathbf{w} \mathbf{x} \mathbf{x}^T - \mathbf{w} \mathbf{x} \mathbf{x}^T \mathbf{w}^T \mathbf{w}) \quad (7)$$

dengan menggunakan operator ekspektasi statistik di kedua ruas, akan menjadi:

$$E[\Delta \mathbf{w}] = \eta (\mathbf{w} E[\mathbf{x} \mathbf{x}^T] - \mathbf{w} E[\mathbf{x} \mathbf{x}^T] \mathbf{w}^T \mathbf{w}) \quad (8)$$

Pada persamaan (8), jika bobot vektor telah mencapai nilai yang stabil (*steady-state*), maka ekspektasi dari bobot vektor *update* adalah nol,  $E[\Delta \mathbf{w}] = 0$  dan ekspektasi produk luar dari vektor *input* adalah matriks kovarian (korelasi):

$$R = E[\mathbf{x} \mathbf{x}^T] \approx \frac{1}{N} \mathbf{X} \mathbf{X}^T, \quad (9)$$

sehingga persamaan (8) dapat dituliskan sebagai berikut:

$$\begin{aligned} \eta (\mathbf{w} E[\mathbf{x} \mathbf{x}^T] - \mathbf{w} E[\mathbf{x} \mathbf{x}^T] \mathbf{w}^T \mathbf{w}) &= 0 \\ \mathbf{w} R &= (\mathbf{w} R \mathbf{w}^T) \mathbf{w} \end{aligned} \quad (10)$$

jika dianggap sebagai skalar, maka:

$$\lambda_1 = \mathbf{w} R \mathbf{w}^T \quad (11)$$

Dengan menggunakan persamaan (11), maka persamaan (10) dapat dituliskan kembali menjadi:

$$\mathbf{w} R = \lambda_1 \mathbf{w} \quad (12)$$

Alternatif lain, jika kita diasumsikan:

$$\mathbf{w}(n) = \pm \mathbf{q}_1^T \text{ jika } n \rightarrow \infty \quad (13)$$

maka :

$$R \mathbf{q}_1 = \lambda_1 \mathbf{q}_1 \text{ dimana } \lambda_1 = \mathbf{q}_1^T R \mathbf{q}_1 \quad (14)$$

Persamaan (14) menunjukkan sepasang vektor eigen  $\mathbf{q}_1$  dan nilai eigen  $\lambda_1$  (nilai terbesar) yang merupakan nilai-nilai karakteristik dari matriks  $R$

dan pada persamaan (13), menegaskan bahwa bobot vektor disatukan menjadi vektor eigen  $\mathbf{q}_1$ .

**B. Penghitungan PCA**

Dari struktur *neuron* tunggal dapat dikembangkan menjadi jaringan (*network*) *p-neuron*, bobot vektor yang berhubungan dengan *neuron* berikutnya mengekstrak vektor eigen berikutnya dari input matriks korelasi.

Aturan pembelajaran yang digunakan, dikenal sebagai aturan Sanger (*Sanger's rule*) atau Generalized Hebbian Algorithm (GHA).

Ide dasar dibalik proses generalisasi jaringan *neuron* Oja adalah penggunaan perbesaran vektor input seperti pada persamaan (6)

Proses *update* bobot pada GHA dapat dijelaskan sebagai berikut:

$$\begin{aligned}
 y_1 &= \mathbf{w}_1 \mathbf{x}, \tilde{\mathbf{x}}_1 = \mathbf{x} - y_1 \mathbf{w}_1^T, \Delta \mathbf{w}_1 = \eta y_1 \tilde{\mathbf{x}}_1^T \\
 y_2 &= \mathbf{w}_2 \mathbf{x}, \tilde{\mathbf{x}}_2 = \tilde{\mathbf{x}}_1 - y_2 \mathbf{w}_2^T, \Delta \mathbf{w}_2 = \eta y_2 \tilde{\mathbf{x}}_2^T \\
 y_j &= \mathbf{w}_j \mathbf{x}, \tilde{\mathbf{x}}_j = \tilde{\mathbf{x}}_{j-1} - y_j \mathbf{w}_j^T, \Delta \mathbf{w}_j = \eta y_j \tilde{\mathbf{x}}_j^T \quad (15)
 \end{aligned}$$

Untuk menuliskan algoritma Generalized Hebbian Learning dalam bentuk matriks, maka dari persamaan (15), perbesaran vektor sinyal *input* dapat dituliskan sebagai berikut:

$$\begin{aligned}
 \tilde{\mathbf{x}}_j^T &= \mathbf{x}^T - [y_1 \dots y_j] \begin{bmatrix} \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_j \end{bmatrix} = \mathbf{x}^T - [y_1 \dots y_j \ 0 \dots 0] \mathbf{W} \\
 &= \mathbf{x}^T - \tilde{\mathbf{y}}_j^T \mathbf{W}
 \end{aligned}$$

dimana  $\tilde{\mathbf{y}}_j^T = [y_1 \dots y_j \ 0 \dots 0]$  adalah *output* dari vektor  $y$  dan komponen  $m - j$  diset menjadi nol. Kemudian untuk *update* bobot *neuron*  $j$  seperti pada persamaan (15) dapat dituliskan kembali menjadi bentuk berikut:

$$\Delta \mathbf{w}_j = \eta (y_j \mathbf{x}^T - y_j \tilde{\mathbf{y}}_j^T \mathbf{W}) \quad (16)$$

Terakhir, pola *update* dari seluruh matriks bobot pada GHA dapat dituliskan dalam bentuk yang lebih sederhana sebagai berikut:

$$\begin{aligned}
 \mathbf{y} &= \mathbf{W} \mathbf{x}, \Delta \mathbf{W} = \eta (\mathbf{y} \mathbf{x}^T - \text{tril}(\mathbf{y} \mathbf{y}^T) \mathbf{W}) \\
 \Delta \mathbf{W} &= \eta (\mathbf{W} \mathbf{x} \mathbf{x}^T - \text{tril}(\mathbf{W} \mathbf{x} \mathbf{W}^T \mathbf{x}^T) \mathbf{W}) \quad (17)
 \end{aligned}$$

dimana  $\text{tril}(\cdot)$  merupakan matriks segitiga bawah yang elemen-elemen di atas diagonal utama diubah menjadi nol. Uraian dari pembuktian vektor-bektor bobot jaringan dapat disatukan menjadi nilai eigen dari input matriks korelasi adalah sebagai berikut:

- a. Dengan menggunakan operator ekspektasi statistik pada kedua ruas dari persamaan *update* bobot (17) dan dengan asumsi dalam keadaan *steady-state* (*update* bernilai nol,  $\Delta \mathbf{W} = 0$ ), maka:

$$0 = E[\mathbf{W} \mathbf{x} \mathbf{x}^T] - E[\text{tril}(\mathbf{W} \mathbf{x} \mathbf{W}^T \mathbf{x}^T) \mathbf{W}]$$

atau dapat ditulis:

$$\mathbf{W} \mathbf{R} = \text{tril}(\mathbf{W} \mathbf{R} \mathbf{W}^T) \mathbf{W} \text{ atau}$$

$$\begin{bmatrix} \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_p \end{bmatrix} \mathbf{R} = \text{tril} \left( \begin{bmatrix} \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_p \end{bmatrix} \mathbf{R} [\mathbf{w}_1^T \dots \mathbf{w}_p^T] \begin{bmatrix} \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_p \end{bmatrix} \right)$$

- b. Jika  $\text{tril}(\mathbf{W} \mathbf{R} \mathbf{W}^T)$  dimisalkan sebagai  $\Lambda$ , maka:

$$\Lambda = \text{tril}(\mathbf{W} \mathbf{R} \mathbf{W}^T) = \begin{bmatrix} \lambda_1 & & \mathbf{0} \\ & \ddots & \\ \{\lambda_{ji}\} & & \lambda_p \end{bmatrix}, \text{dimana } \lambda_{ji} = \mathbf{w}_j \mathbf{R} \mathbf{w}_i^T$$

- c. Sehingga secara lengkap dapat dituliskan sebagai berikut:

$$\begin{bmatrix} \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_p \end{bmatrix} \mathbf{R} = \begin{bmatrix} \lambda_1 & & \mathbf{0} \\ & \ddots & \\ \{\lambda_{ji}\} & & \lambda_p \end{bmatrix} \begin{bmatrix} \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_p \end{bmatrix} \quad (18)$$

atau:

$$\mathbf{W} \mathbf{R} = \Lambda \mathbf{W} \quad (19)$$

Baris pertama dari persamaan (18), jika dituliskan terpisah akan seperti:

$$\mathbf{w}_1 \mathbf{R} = \lambda_1 \mathbf{w}_1, \quad \text{dimana } \lambda_1 = \mathbf{w}_1 \mathbf{R} \mathbf{w}_1^T$$

bentuk di atas sama dengan aturan Oja (lihat persamaan 10 dan 11).

**2.3.4 Back Propagation Network**

Ide dasar dari BPN yang termasuk dalam *Supervised Learning* (pembelajaran dengan pengawasan) ini adalah secara efisien menghitung penurunan parsial dari fungsi aproksimasi  $F(\mathbf{w}; \mathbf{x})$  yang diterapkan pada jaringan dengan elemen-elemen:

- a.  $\mathbf{w}$ , vektor bobot yang nilainya dapat disesuaikan, terhadap
  - b.  $\mathbf{x}$ , nilai-nilai dalam vektor input.
- yang di dalamnya terdapat kekuatan perhitungan menggunakan algoritma propagasi balik.[3]

Pertama, BPN akan dilatih dengan data latih yaitu berupa ciri-ciri penting dari gambar manusia telanjang termasuk ciri-ciri gambar yang bukan kategori manusia telanjang. Hasil dari proses pelatihan adalah mendapatkan bobot-bobot sinaptik yang stabil pada BPN. Kedua, adalah BPN digunakan untuk melakukan pengujian/klasifikasi apakah gambar input terdapat manusia telanjang atau tidak. Dalam BPN ini terdapat istilah-istilah sebagai berikut [3]:

1. *Epoch*, setiap satu proses pelatihan secara lengkap dengan seluruh data pelatihan selama proses pelatihan berlangsung disebut 1 *epoch*.
2. *Error signal* (E), adalah sinyal *error* yang berasal dari *neuron output* JST dan berpropagasi balik (layer demi layer). *Error signal* sama dengan selisih yang muncul antara matriks target dengan matriks *output*.
3. *Sum Square Error* (SSE), merupakan jumlah kuadrat dari *Error signal* yang muncul pada setiap *epoch*.

Berikut adalah algoritma untuk melakukan proses pelatihan BPN:

1. Definisi masalah, misal **P** adalah matriks *input* dan **T** adalah matriks target.
2. Inisialisasi, menetapkan bentuk jaringan dan menetapkan nilai-nilai bobot sinaptik (*W1* dan *W2*) dan *learning rate* (*lr*). Pemilihan nilai-nilai awal yang baik pada *W1* dan *W2* dapat meningkatkan performansi dari jaringan yang dibangun. Penentuan nilai awal dari setiap parameter diberikan nilai *random* yang terdistribusi normal dalam rentang yang kecil (biasanya antara -1 dan +1). *Learning rate* menentukan seberapa cepat bobot-bobot sinaptik pada *W1* dan *W2* akan memenuhi kondisi stabil. Semakin kecil nilai yang kita ambil, maka semakin kecil pula perubahan yang terjadi pada bobot-bobot sinaptik.
3. Pelatihan, memberikan data pelatihan pada JST hingga *epoch* tertentu. Untuk setiap data pelatihan yang diberikan, dilakukan perhitungan maju (*forward computation*) dan perhitungan mundur (*backward computation*) sebagai berikut:

a. Perhitungan Maju (*Forward Computation*)

Keluaran dari *hidden layer* dan *output layer*:

$$A1 = \frac{1}{1 + e^{-\sum_{i=1}^m P_i * W1_{ij}}}; \text{ hidden layer}$$

$$A2 = \frac{1}{1 + e^{-\sum_{j=1}^n A1_j * W2_{kj}}}; \text{ output layer}$$

Galat/*Error Signal* (E) dan *Sum Square Error* (SSE) didefinisikan sebagai berikut:  $E = T - A2$

$$SSE = \sum E^2$$

b. Perhitungan Mundur (*Backward Computation*)

Perhitungan mundur dilakukan untuk memperbaiki bobot-bobot sinaptik pada jaringan BPN. Bobot sinaptik untuk *hidden layer* *W1* dan *layer output* *W2*, propagasi balik didefinisikan sebagai berikut :

$$\begin{aligned} D2 &= A2 * (1 - A2) * E \\ dW2 &= dW2 + (lr * D2 * A1) \\ D1 &= A1 * (1 - A1) * (W2 * D2) \\ dW1 &= dW1 + (lr * D1 * P) \\ W2 &= W2 + dW2 \\ W1 &= W1 + dW1 \end{aligned}$$

Kedua langkah di atas adalah untuk satu kali siklus pelatihan (1 *epoch*).

4. Iterasi, proses pelatihan diulang sampai jumlah *epoch* tertentu atau telah mencapai SSE yang diinginkan.
5. Hasil akhir pelatihan jaringan adalah didapatkannya bobot-bobot *W1* dan *W2* yang kemudian disimpan untuk proses pengujian jaringan.

Pada prakteknya, perancangan BPN bergantung pada masalah yang akan diselesaikan. Jika data *input* berdimensi besar atau jumlah kelas *output* yang diinginkan besar, maka diperlukan lebih banyak jumlah *node* pada *hidden layer* atau diperlukan lebih dari satu *hidden layer*, tetapi ada batas optimal untuk kedua parameter tersebut.

### 3. Perancangan Sistem

#### 3.1 Deskripsi Sistem

Sistem ini merupakan suatu *filter* pornografi yang dikhususkan untuk menangani gambar manusia telanjang dan masih terbatas untuk memeriksa 1 (satu) objek dalam gambar.

Langkah 1: Menerima gambar masukan dan melakukan proses pemisahan warna kulit dan bukan kulit yang terdapat dalam gambar. Warna bukan kulit akan diubah ke warna hitam.

Langkah 2: Ekstraksi ciri menggunakan metode *Hebbian Learning*.

Langkah 3: Proses identifikasi menggunakan *Back Propagation Network*, tahap ini terbagi menjadi dua bagian, yaitu tahap pelatihan dan tahap pengujian.

##### 3.1.1 Tahap Pelatihan

Tahap pelatihan digunakan untuk melatih jaringan syaraf tiruan dengan masukan berupa bobot-bobot vektor dari proses *Hebbian Learning*. Pada proses

pelatihan ini, bobot-bobot vektor yang sudah stabil akan disimpan ke dalam *library* (kamus) dan siap digunakan untuk proses pengujian.



Gambar 3. Diagram Blok Proses Pelatihan

### 3.1.2 Tahap Pengujian

Dalam tahap pengujian ini, jaringan syaraf tiruan *Back Propagation Network* akan menggunakan bobot-bobot vektor yang telah disimpan dalam *library* untuk mengklasifikasikan bobot-bobot vektor input dari proses ekstraksi ciri.



Gambar 4. Diagram Blok Proses Pengujian

## 3.2 Strategi Pengujian

Pengujian pertama kali dilakukan pada metode Hebbian *Learning*. Konfigurasi pengujian adalah sebagai berikut:

1. Pada proses pelatihan Hebbian *Learning*, jumlah *hidden neuron* diubah-ubah mulai dari 30 hingga 60 *neuron*.
2. Dari setiap perubahan jumlah *neuron*, hasil pelatihan yang didapatkan kemudian dilatihkan pada BPN dengan konfigurasi standar, yakni : jumlah *hidden-neuron* 30, *learning rate* = 0,5, *threshold* = 0,5.
3. Setelah dilatihkan pada BPN, maka proses selanjutnya adalah pengujian menggunakan data uji sebanyak 610 citra.
4. Setelah didapatkan hasil yang terbaik, maka pada konfigurasi BPN dilakukan perubahan-perubahan untuk mendapatkan hasil yang lebih baik lagi.

## 3.3 Pemilihan data

### 3.3.1 Data Latih

Data latih *non-nude* yang dipilih adalah yang mewakili kasus-kasus sebagai berikut:

1. Citra manusia berpakaian
2. Citra manusia dengan pakaian yang tipis
3. Citra manusia dengan warna latar belakang menyerupai warna kulit
4. Citra manusia dengan tingkat pencahayaan yang berbeda-beda
5. Citra pemandangan

Sedangkan data latih untuk kategori *nude*, adalah yang mewakili kasus-kasus sebagai berikut:

1. Citra manusia telanjang dengan warna kulit yang berbeda-beda (ras *Caucasian*, *Asian*, *Black*)
2. Citra manusia telanjang dengan sebagian tubuhnya tertutup oleh benda (kain, kayu, dll)
3. Citra manusia telanjang yang memiliki warna latar belakang menyerupai warna kulit.
4. Citra manusia telanjang dengan berbagai macam pose
5. Citra manusia telanjang dengan tingkat pencahayaan yang berbeda-beda.

### 3.3.2 Data Uji

Dalam pengujian kasus-kasus *miss-detection*, digunakan data citra manusia telanjang dengan berbagai macam variasi latar belakang, tingkat pencahayaan yang ekstrim, atau citra yang sebagian tubuhnya terhalangi oleh pakaian/benda dan posisi objek manusia tidak berada di tengah-tengah.

Pada pengujian kasus-kasus *false-alarm*, digunakan citra manusia mengenakan pakaian yang warnanya menyerupai warna kulit manusia, latar belakang yang menyerupai area kulit, dan citra dengan objek-objek lain yang warnanya mirip dengan warna kulit.

Dari seluruh data yang sudah berhasil dikumpulkan, dikategorikan menjadi tiga kelompok besar, yakni:

1. Foto-foto natural: binatang, pemandangan, tempat-tempat wisata, dan lain-lain;
2. Foto-foto dari majalah;
3. Kumpulan citra-citra manusia telanjang dari beberapa *web site* di internet.

## 4. Hasil

Pengujian dilakukan terhadap 610 gambar yang terdiri dari :

1. 305 citra *nude*, dan
2. 305 citra *non-nude*.

Dalam proses pengujian ini akan dicari akurasi (prosentase keberhasilan secara menyeluruh), diperoleh dengan cara: jumlah citra yang diidentifikasi dengan benar dibagi dengan jumlah seluruh citra uji dan dikali dengan 100%.

$$\text{Akurasi} = \frac{\text{jumlah citra dikenali dengan benar}}{610} \times 100\%$$

### 4.1 Hasil Pengujian Hebbian *Learning*

Dimensi data masukan proses Hebbian *Learning* ini sebesar 60.000 x 40. Angka 60.000 diperoleh dari dimensi citra masukan 300 x 200 piksel, dan 40 adalah jumlah data yang dilatihkan. Pengujian metode ini dilakukan dengan melakukan perubahan jumlah *neuron* yang digunakan mulai dari

30 hingga 60. Jumlah *neuron* ini menunjukkan besarnya dimensi data setelah proses Hebbian *Learning*, jika digunakan *neuron* sejumlah 30, maka dimensi keluaran dari proses ini adalah: 30x40. Jadi tujuan dari proses Hebbian *Learning* ini adalah mereduksi besarnya dimensi data masukan sehingga didapatkan komponen-komponen utamanya saja.

Langkah pertama proses pengujian adalah dengan menggunakan *neuron* sejumlah 30, 40, 50 dan 60. Dari keempat jumlah *neuron* tersebut didapatkan hasil yang ditunjukkan pada tabel 1.

Dari hasil pengujian yang telah dilakukan, dapat dilihat bahwa dengan jumlah *neuron* sebanyak 40 memiliki prosentase kebenaran paling tinggi yakni **74,9180 %**. Jadi 40 komponen yang didapatkan sudah cukup mewakili komponen sejumlah 60.000 masih mampu mengidentifikasi citra-citra uji lebih baik daripada jumlah *neuron* yang lainnya. Hasil pengujian kemudian diarahkan pada jumlah *neuron* antara 30 sampai 50.

TABEL 1  
HASIL PROSENTASE KEBERHASILAN IDENTIFIKASI  
DENGAN JUMLAH *NEURON* : 30,40,50,60

Jumlah neuron	Benar	Salah	Akurasi (%)
30	348	262	65.0491
<b>40</b>	<b>406</b>	<b>204</b>	<b>74.9180</b>
50	355	255	69.1967
60	344	266	66.3934

Langkah kedua pengujian adalah uji sensitivitas dengan mencari kemungkinan jumlah *neuron* yang paling tepat dan mampu menghasilkan prosentase keberhasilan yang lebih tinggi. Dari percobaan yang telah dilakukan, maka dapat diamati bahwa jumlah *neuron* yang dapat memberikan prosentase kebenaran tertinggi adalah **40**.

TABEL 2  
HASIL UJI SENSITIVITAS JUMLAH *NEURON*

Jumlah neuron	Benar	Salah	Akurasi (%)
31	350	260	65.3770
33	378	232	69.9672
35	293	317	56.0327
37	354	256	66.0327
39	294	316	56.1967
<b>40</b>	<b>406</b>	<b>204</b>	<b>74.9180</b>
41	313	297	59.3114
43	328	282	61.7704
45	324	286	61.1147
47	344	266	64.3934
49	355	255	66.1967
50	355	255	69.1967
51	265	345	54.4426

## 4.2 Hasil Pengujian *Back Propagation Network*

### 4.2.1 Mencari Jumlah *Hidden-Neuron* yang Optimal

Langkah pengujian BPN yang pertama adalah mencari jumlah *hidden-neuron* yang optimal baik secara prosentase kebenaran. Jumlah *hidden-neuron* yang digunakan akan dikurangi satu per satu dan digunakan untuk pengujian menggunakan data latih (data latih yang digunakan adalah 20 data pertama dari masing-masing golongan citra). Pada proses pengujian ini didapatkan prosentase keberhasilan paling tinggi dihasilkan dengan menggunakan 30, 29 dan 28 *neuron* pada *hidden-layer* atau mengurangi 0 sampai 2 buah *hidden-neuron*. Hal ini menunjukkan bahwa meskipun dengan 28 *hidden-neuron* pada BPN dapat digunakan untuk mengidentifikasi gambar manusia telanjang secara optimal, dan lebih menghemat penggunaan memori dalam proses komputasi.

### 4.2.2 Mencari *Threshold* yang Optimal

Langkah kedua pengujian BPN adalah mencari nilai *threshold* yang optimal terhadap prosentase kebenaran. Nilai *threshold* didapatkan dengan cara melakukan percobaan pengujian menggunakan seluruh data uji yakni 610 citra. Dari percobaan didapatkan nilai *threshold* yang paling optimal adalah **0,4**. Nilai ini menunjukkan bahwa jika nilai pemetaan sebuah citra ternyata lebih besar atau sama dengan 0,4 ( $\geq 0,4$ ) maka akan dianggap sebagai citra *nude*.

### 4.2.3 Waktu Proses Identifikasi

Total waktu pengenalan sebanyak 610 citra yang sudah tersegmentasi adalah 8 menit 8 detik (488 detik), jadi rata-rata identifikasi tiap satu citra memerlukan waktu 0,8 detik. Sedangkan waktu yang dibutuhkan untuk melakukan proses segmentasi pada satu citra adalah 7 detik. Jadi waktu total yang dibutuhkan untuk mengenali satu buah citra sebelum disegmentasi adalah 7,8 detik.

## 4.3 Rangkuman Hasil Pengujian

Setelah dilakukan berbagai pengujian terhadap sistem pengenalan manusia telanjang ini, maka dapat dirangkum sebagai berikut:

1. Prosentase keberhasilan identifikasi citra *nude* sebagai citra *nude* mencapai **84,5901%** (284 citra dari 305 *nude*), sedangkan prosentase keberhasilan identifikasi citra *non-nude* sebagai citra *non-nude* mencapai **65,9016%** (201 citra dari 305 citra *non-nude*).
2. *Miss-detection*. Pada gambar 5 citra masukan memiliki pencahayaan yang berlebihan, sehingga hasil segmentasi tidak menyerupai tubuh manusia dan dikenali sebagai citra *non-nude*.

Kasus seperti ini mencapai **15,4098%** (47 citra dari 305 citra *nude*).

3. *False-alarm*. Gambar 6, menunjukkan kesalahan dalam mengenali citra *non-nude*, gambar ini menunjukkan contoh *false-alarm*, dimana citra *non-nude* dikenali sebagai citra *nude*. Hal ini terjadi karena faktor pencahayaan yang cukup ekstrim sehingga hampir seluruhnya menyerupai warna kulit. Prosentase kasus seperti ini mencapai **34,0983%** (104 citra dari 305 *non-nude*)

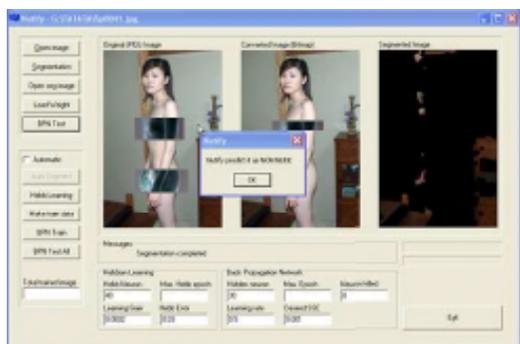
## 5. Kesimpulan

1. Identifikasi gambar manusia telanjang menggunakan metode *Hebbian Learning* dan *Back Propagation Network (BPN)* didapatkan data sebagai berikut:
  - a. *Threshold* terbaik yang digunakan adalah = **0,4**.
  - b. Prosentase pengenalan citra *nude* sebagai *nude* adalah **84,5901 % (258/305)**
  - c. Prosentase pengenalan citra *non-nude* sebagai *non-nude* adalah **65,9016 % (201/305)**
  - d. Citra *nude* dikenali sebagai *non-nude* (*miss-detection*), sebesar **15,4098 % (47/305)**.
  - e. Citra *non-nude* dikenali sebagai *nude* (*false-alarm*) sebesar **34,0983 % (104/305)**.
  - f. Prosentase pengenalan secara keseluruhan adalah **75,2459 % (459/610)**.

2. Jumlah *hidden-neuron* yang digunakan pada BPN sejumlah 28 memberikan prosentase pengenalan yang paling baik, dari jumlah total 30 neuron. Semakin kecil jumlah *hidden-neuron*, akan semakin menghemat pemakaian memori.
3. Waktu identifikasi sebuah citra masukan yang sudah tersegmentasi, mencapai 0,8 detik. Sedangkan untuk citra yang belum tersegmentasi mencapai 7,8 detik karena waktu yang diperlukan untuk melakukan segmentasi terhadap satu citra adalah 7 detik.
4. Jumlah *neuron* pada *Hebbian Learning* sebanyak 40 buah yang menghasilkan vektor eigen dengan dimensi 40x1 sudah cukup mewakili sebuah citra berwarna 24 bit berukuran 300x200 piksel.

## Daftar Pustaka

- [1] Fleck, M. M., Forsyth, D. A., Bregler, C., *Finding Naked People*, European Conference on Computer Vision, vol. II, New York: Springer-Verlag, 1996, hal. 593-602.
- [2] Gonzales, Rafael C., *Digital Image Processing*, Addison-Wesley Publishing Company, Inc, USA, 1993
- [3] Simon Haykin., *Neural Networks – a Comprehensive Foundation* 2<sup>nd</sup> edition, New Jersey: Prentice Hall, 1999, ISBN 0-13-273350-1.
- [4] Rossotti, Hazel., *Colour : Why the world isn't Grey ?*, Prinxteton University Press. Princeton, NJ, 1983



Gambar 5. Miss Detection



Gambar 6. False Alarm