BAB I

PENDAHULUAN

1.1 Latar Belakang

Berdasarkan statistik dari Badan Pekerja Amerika Serikat, terjadi luka fatal sebanyak 393 kasus yang disebabkan oleh kejatuhan benda di tempat kerja. Diantaranya yaitu kasus seorang pekerja di galangan kapal sedang mengelas pelat baja untuk membangun dek kapal ketika las tiba-tiba pecah, menyebabkan pelat tersebut terlepas. Salah satu pelat baja yang berat ini menghantam sisi topi keras pria itu dengan sangat kuat hingga menembus kulit topinya. Pekerja itu mengalami cedera; namun, topi keras itu secara dramatis mengurangi keparahan pukulan itu. Sejauh ini, dia masih bisa bekerja. Kasus kedua datang dari seorang subkontraktor listrik di Portland sedang bekerja pada ketinggian sekitar 50 meter di atas tanah ketika kepalanya dihantam oleh pipa plastik 6,6 lb, jatuh dari ketinggian 20 meter di atasnya. Berkat topi keras yang dia kenakan, pekerja tersebut selamat tanpa cedera yang melemahkan atau fatal. Lalu kasus ketiga datang dari seorang pekerja sedang bersiap untuk meledakkan jembatan terdekat di Buffalo. Dia tidak terkena bahaya di atas kepala atau lalu lintas (tidak ada bahaya yang terlihat) tetapi masih mengenakan topi keras seperti yang diarahkan oleh panduan praktik yang aman dan akal sehat. Tiba-tiba, palka baja seberat 100 pon meledakkan bagian atas peralatan peledakan tekanan, yang berada beberapa kaki dari pekerja, dan mengenai kepalanya tepat di depan. Dampak dari palka baja membuat pekerja tersebut jatuh ke tanah. Pekerja tersebut dibawa ke rumah sakit terdekat, tetapi tidak ditemukan luka serius. Fakta ini bisa dicegah jika semua pihak memastikan bahwa perusahaan memastikan pekerja memakai *hardhat* atau helm keselamatan kerja [7].

Akan tetapi deteksi manual memiliki beberapa kelemahan karena staf keselamatan tidak dapat memastikan pekerja migas mengenakan helm keselamatan kerja karena mereka tidak dapat memantau pekerja migas setiap hari dan setiap jam. *Computer vision* terbukti efektif mengambil data yang relevan dengan cepat dan mudah.

Beberapa penelitian lainnya di bidang deteksi helm keselamatan kerja telah dilakukan. Shresta, dkk (2016) mengidentifikasi apakah pekerja memakai helm keselamatan kerja atau tidak pada industri konstruks bangunan. Metode yang digunakan antara lain deteksi tepi dan segmentasi menggunakan *Haar-like feature* untuk deteksi wajah dan deteksi berdasar warna untuk deteksi helm. Namun tidak dipaparkan evaluasi dari algoritma yang dikembangkan [26]. *Haar-like feature* merupakan salah satu jenis *Cascade Classifier*. Metode ini juga dibandingkan dengan metode *Cascade Classifier* lainnya seperti *Local Binary Pattern* (LBP) dan *Histogram of Oriented Gradients* (HOG) menggunakan 75 dataset gambar positif (pekerja memakai helm) dan 164 dataset negatif (pekerja tanpa helm). Hasil dari penelitian ini menunjukkan bahwa Haar dan LBP lebih banyak salah mendeteksi pada semua dataset *testing*. Sedangkan *Histogram of Oriented Gradients* (HOG) dapat mendeskripsikan bentuk lingkaran helm secara akurat [19]. Penelitian lainnya yang mendeteksi pengendara yang memakai helm dan pengendara tanpa helm dalam bentuk video menggunakan *pre-processing* Median filter dan *thresholding* serta *Local Binary Pattern* (LBP) dengan sebelumnya dataset *training* dipotong hanya 20% area teratas *bounding boxes* manusia. Area ini dinamakan *Region of Interest* (ROI). Namun tidak dijelaskan hasil dan evaluasi dari penelitian ini [4]. Metode *Region of Interest* (ROI) juga diterapkan dengan pengambilan area 1/3 area teratas dan 1/2 lebar *bounding box* objek manusia. Menggunakan metode Haar sebagai descriptor, studi ini mencapai tingkat akurasi 75% dan 74% berturut-turut menggunakan helm dan tanpa helm dalam deteksi helm melalui CCTV [32]. Seperti yang sudah disebutkan sebelumnya, metode ekstraksi fitur *Histogram of Oriented Gradients* (HOG) dapat mengambil bentuk lingkaran dengan baik jika digabungkan dengan metode *ViBe Background Modelling* dan metode klasifikasi *Support Vector Machine* (SVM) mencapai tingkat akurasi lebih dari 80% pada deteksi helm secara *real-time* pada video [16, 17].

Tabel 1.1 Perbandingan referensi dan metode yang diusulkan

	(Rubaiyat, et.al, 2016)	**(Vishnu, et.al, 2017)**	**Metode yang diusulkan**

Jenis objek	Helm	Pengendara motor, kepala	**Helm**
Metode	- GMM untuk segmentasi citra - DCT + HOG - Linear SVM - Ekstraksi fitur berdasarkan warna - *Circular Hough Transform* (CHT)	- GMM untuk segmentasi citra - CNN for human detection	***-Generalized Hough Transform*** ***-Autoencoder*** ***-Support Vector Machine* (SVM)**
Evaluasi	Tingkat akurasi 91,93%	Tingkat akurasi HOG = 57% dan CNN = 87%	
Informasi dataset	1000 gambar untuk data training dan 200 gambar untuk data testing	Dua jam video (25 fps) menjadi dataset training dan testing (3 : 1)	

Studi tentang deteksi helm juga dilakukan oleh Rubaiyat, dkk (2016). Metode yang digunakan yakni *Gaussian Mixture Model* (GMM) untuk segmentasi citra, *discrete cosine* transform (DCT) dan *Histogram of Oriented Gradients* (HOG) sebagai deskriptor, ekstraksi fitur berdasar warna, *Circular Hough Transform* (CHT) dan *Support Vector Machine* (SVM) linear untuk klasifikasi. Tingkat akurasi yang didapatkan adalah 91,93% untuk 1000 dataset *training* dan 200 dataset *testing* [25].

Jenis ekstraksi fitur *hough transform* lainnya yaitu *Generalized Hough Transform* (GHT) diterapkan dalam penelitian deteksi objek Yousef, dkk (2016). Model pertama yang menggunakan (*Scale-Invariant Feature Transform*) SIFT dan GHT lebih baik daripada model kedua yang menggunakan SIFT dan *Random Sample Consensus* (RANSAC) [33]. Metode GHT yang juga digunakan dalam algoritma pengenal karakter huruf Arab menghasilkan rerata tingkat keberhasilan sebesar 92% pada tiga eksperimen dilakukan dengan jumlah dataset yang digunakan berturut-turut 166.873 buah sampel font *Arabic Transparent*, 234.868 buah sampel dari tiga font Arab, dan 1000 buah sampel font Arab campuran dengan ukuran yang berbeda [28].

Untuk mengurangi dimensi ruang parameter, peneliti harus melakukan *voting* multi-tahap. Sebagai contoh, dalam dua tahap GHT oleh Tsai [29] setiap titik batas dalam gambar dideskripsikan oleh tiga fitur yaitu *concavity*, radius dan arah normal dari segmen kurva di lingkungan titik. Tahap pertama dari proses *voting* menentukan sudut rotasi dari objek w.r.t dengan mencocokkan titik-titik yang memiliki kecekungan dan radius yang sama. Tahap kedua kemudian menentukan pusat objek sensorik dengan mencocokkan titik-titik yang memiliki jari-jari, kecekungan, dan sudut rotasi yang sama [21].

Penelitian Vishnu, dkk (2017) menunjukkan bahwa penggunaan *neural network* dapat meningkatkan tingkat akurasi deteksi helm [30]. Selain itu, implementasi *hough transform* dengan *neural network* dalam [2, 5, 17] dapat meningkatkan kecepatan deteksi objek. Pendekatan ini menerima koordinat gambar sebagai masukan dan mempelajari bentuk parametrik garis secara adaptif [19]. Salah satu *neural network* yang disebut dengan *Autoencoder* menghasilkan *error rate* yang lebih rendah apabila dibandingkan dengan metode reduksi dimensionalitas lainnya seperti *Linear Discriminant Analysis* (LDA), *Principal Component Analysis* (PCA), dan *Marginal Fisher Analysis* (MFA) pada aplikasi pengenalan wajah dan klasifikasi digit MNIST [32, 1].

Penelitian yang telah dilakukan sebelumnya mencapai tingkat akurasi 63% pada klasifikasi pengguna helm dan tidak menggunakan SVM dengan sebelumnya dilakukan deteksi objek manusia sehingga setelah proses training 111 gambar berlabel “menggunakan helm” dan 103 gambar berlabel “tidak menggunakan helm” yang telah dilakukan *preprocessing* menggunakan *Canny edge detection* dan *Histogram of Gradients* (HOG), diujikan pada 100 gambar pejalan kaki yang menggunakan maupun tidak menggunakan helm dengan perbandingan 2:1 berupa area *bounding boxes* hasil deteksi objek manusia menggunakan YOLO. Usaha untuk mendeteksi banyak objek helm secara bersamaan belum dicapai pada penelitian tersebut dan tingkat akurasi yang dihasilkan masih rendah dikarenakan pelabelan dilakukan secara otomatis dan pada penelitian yang penulis usulkan akan dilakukan pelabelan secara manual [20].

Berdasarkan penelitian-penelitian yang telah dilakukan, penulis tertarik melakukan penelitian yang menerapkan *Computer Vision* berjudul : "Deteksi Helm Keselamatan Kerja Berbasis Citra Menggunakan *Autoencoder* dan *Support Vector Machine* Dibandingkan dengan *Generalized Hough Transform*". Adapun pada deteksi helm keselamatan terdiri dari dua tahap yaitu ekstraksi fitur dan klasifikasi objek. Ekstraksi fitur dilakukan menggunakan *autoencoder* sedangkan klasifikasi dilakukan dengan menggunakan *Support Vector Machine* (SVM) dan *Generalized Hough Transform* (GHT). Untuk menggunakan GHT diperlukan deteksi tepi terlebih dahulu menggunakan deteksi tepi Canny.

1.2 Rumusan Masalah

1. Bagaimana ekstraksi fitur menggunakan *Autoencoder* dan deteksi tepi Canny pada deteksi helm keselamatan kerja berbasis citra?
2. Bagaimana klasifikasi objek *hardhat* dan bukan *hardhat* menggunakan *Support Vector Machine (SVM) dan Generalized Hough Transform (GHT)* pada deteksi helm keselamatan kerja berbasis citra?
3. Bagaimana evaluasi performa algoritma terhadap dataset?

1.3 Tujuan

1. Menjelaskan ekstraksi fitur menggunakan Autoencoder dan deteksi tepi Canny pada deteksi helm keselamatan kerja berbasis citra.
2. Menjelaskan klasifikasi objek *hardhat* dan bukan *hardhat* menggunakan *Support Vector Machine (SVM) dan Generalized Hough Transform (GHT)* pada deteksi helm keselamatan kerja berbasis citra.
3. Menjelaskan evaluasi performa algoritma terhadap dataset.

1.4 Manfaat

1. Bagi Mahasiswa

   Menambah ilmu pengetahuan secara teoritis tentang *Computer Vision.*
2. Bagi Peneliti

   Memberikan kontribusi untuk bahan diskusi pada deteksi helm keselamatan kerja berbasis citra dengan *Computer Vision.*

3. Bagi Umum

   Memberikan informasi tentang penerapan pengolahan citra pada sistem deteksi helm keselamatan kerja berbasis citra.

BAB II

LANDASAN TEORI

2.1 *Hardhat*

Hardhat merupakan helm keselamatan kerja yang biasa digunakan oleh pekerja konstruksi dan migas untuk melindungi kepala dari terluka akibat kejatuhan benda berat dan sebagainya. Pita suspensi di dalam helm menyebarkan berat helm dan kekuatan benturan apapun di atas kepala. Suspensi juga menyediakan ruang sekitar 30 mm (1,2 inci) antara cangkang helm dan kepala pemakainya, sehingga jika ada benda yang mengenai cangkang, dampaknya kecil kemungkinannya untuk ditransmisikan langsung ke tengkorak. Beberapa cangkang helm memiliki punggung penguat garis tengah untuk meningkatkan ketahanan benturan [10].

2.2 *Computer Vision*

Computer vision merupakan sebuah bidang sains yang mengekstraksi informasi dari citra digital. Jenis informasi yang didapatkan dari citra beragam : identifikasi, pengukuran ruang untuk navigasi, atau aplikasi lainnya. Definisi lain *computer vision* yakni algoritma yang dapat memahami konten citra dan menggunakannya untuk keperluan lain. Sejarah awalnya adalah proyek liburan mahasiswa sarjana *Massachusetts Institute of Technology* (MIT) tahun 1966 yang diperkirakan dapat selesai selama liburan namun ternyata sampai sekarang sudah 50 tahun lamanya bidang sains ini masih menarik untuk diteliti [13].

Adapun beberapa penerapan *computer vision* antara lain : *Optical character recognition* (OCR), inspeksi mesin, retail, informatika kedokteran, keselamatan otomotif, *3D model building*, *match move*, pengawasan, biometrika, pengenalan sidik jari, deteksi wajah, dan otentikasi visual [27].

2.3 *Generalized Hough Transform* (GHT)

Kali pertama diperkenalkan untuk mendeteksi garis pada gambar, *hough transform* dikembangkan untuk ektraksi fitur geometri lainnya seperti lingkaran dan

elips. *Generalized Hough Transfom* (GHT) digunakan untuk mencari posisi model dua dimensi pada gambar yang memiliki derau dengan model-model yang dipetakan menjadi citra *scene* dengan transformasi yang dibentuk dari translasi, rotasi, dan perubahan skala melalui tahap *preprocessing* model secara *off-line* dan tahap pengenalan objek secara *on-line*. Orientasi dan skala objek diasumsikan tetap, sehingga model akan ditranslasikan di dalam citra *scene* relatif terhadap posisi pada citra model. Pada deskipsi berikut, $\boldsymbol{x}$ digunakan sebagai titik *scene* berbeda dari $\boldsymbol{x}^m$ yang merupakan titik model.

Berikut ini tahap *preprocessing* secara *off-line* :

1. Untuk setiap point $\boldsymbol{x}^m$ pada batas model hitung $\theta(\boldsymbol{x}^m)$, sudut tangen pada $\boldsymbol{x}^m$
2. Ambil sembarang titik perwakilan $\boldsymbol{p}_0$(dengan koordinat model $\boldsymbol{x}_0^m$; pada *scene* titik ini diberi nama $\boldsymbol{x}_0$, titik yang dicari) dan hitung posisi relatifnya terhadap semua titik batas; dengan begitu, untuk setiap titik model kurva $\boldsymbol{x}^m$ hitung $\boldsymbol{r} = \boldsymbol{x}_0^m - \boldsymbol{x}^m$. Catatan : diketahui titik batas *scene* manapun $\boldsymbol{x}$ di dalam citra yang telah ditranslasi dan vektor referensi *r* dihitung untuk titik ini, lokasi titik referensi $\boldsymbol{p}_0$ pada citra ini adalah $\boldsymbol{x}_0 = \boldsymbol{x} + \boldsymbol{r}$.
3. Buat tabel *lookup* yang berisi semua pasangan sudut tangen dengan vektor-vektor referensi yang bersesuaian,

$$\{(\theta(x^m), (x_0^m - x^m))\}.$$

Tabel diindeks dengan sudut diskrit θ. Karena invarian terhadap translasi bidang citra, θ berfungsi sebagai sebuah indeks menjadi tabel selama tahap pengenalan, untuk mengekstraksi vektor-vektor referensi. Tabel ini disebut dengan tabel R.

Tahap pengenalan objek GHT mirip dengan transformasi *Hough* standar : koordinat titik referensi $\boldsymbol{p}_0$ pada citra yang ditranslasi (*scene*) adalah parameter yang tidak diketahui. Sebuah *array* akumulator dibuat sebagai himpunan diskrit koordinat *x* dan *y* $\boldsymbol{p}_0$ yang mungkin. Sekarang *x* pada *scene* yang bisa bagian dari batas model

atau bisa juga tidak dapat menggunakan sudut tangennya $\theta(\boldsymbol{x})$ untuk diindeks ke dalam tabel R, mengekstraksi vektor-vektor referensi yang sudah dipasangkan dengan $\theta(\boldsymbol{x})$ (dengan harapan hanya sedikit), menghitung koordinat $\boldsymbol{p}_0$ pada *scene*, dan melakukan *voting* di dalam *array*. Jadi, untuk kasus orientasi dan skala yang tetap, tahap pengenalan objek adalah sebagai berikut.

1. Inisialisasi *array* akumulator $A(x_0)$ lokasi titik referensi yang mungkin menjadi nol.
2. Untuk setiap titik *edge* $\boldsymbol{x}$, *vote* lokasi-lokasi $\boldsymbol{p}_0$ yang mungkin :
   Untuk setiap entri tabel $\boldsymbol{r}$ pada index $\theta(\boldsymbol{x})$, tambah sel *array* akumulator
   $$A(\boldsymbol{x}_0) \leftarrow A(\boldsymbol{x}_0) + 1$$
3. Nilai maksima pada A berkorespondensi dengan lokasi-lokasi titik referensi yang mungkin

Setelah lokasi-lokasi titik referensi model yang potensial pada *scene* diidentifikasi, kemudian dapat digunakan untuk memetakan semua titik kurva terhadap *scene* dan memverifikasi kandidat satu per satu. Ketika banyaknya objek pre-model dicari, setiap objek model yang mungkin dapat dicari juga secara independen secara urut, namun pencarian ini dapat dikerjakan secara parallel dengan membuat *array* akumulator untuk setiap model yang diketahui ini dan mengulangi tahapan *voting* sekali untuk setiap titik *scene*, untuk setiap model.

Rotasi dan penskalaan termasuk dalam GHT dengan mengembangkan *array* akumulator dan membuat tabel R yang banyak untuk setiap model : satu tabel untuk setiap sudut rotasi yang dikuantisasi dan untuk setiap faktor skala. Sebetulnya, tabel tambahan tidak harus dibuat secara eksplisit; komputasi sederhana selama tahap pengenalan secara otomatis menurunkan sebuah entri pada tabel yang terskala dan dirotasi manapun dari entri yang berkorespondensi pada tabel R tunggal yang sebetulnya disimpan (sehingga setiap nilai yang disimpan menghasilkan *vote* banyak). Untuk menggambarkan perhitungan ini lebih sederhana menggunakan koordinat polar vektor referensi, contohnya $\boldsymbol{r} = (\boldsymbol{r}, \boldsymbol{\alpha})$**.** Untuk pengenalan dalam hal transformasi kesamaan melalui tahapan berikut.

1. Inisialisasi *array* akumulator 4-D $A(x_0, \beta, s)$ lokasi *scene* $\boldsymbol{x}_0$ yang mungkin dari titik referensi $\boldsymbol{p}_0$ untuk apapun sudut rotasi β yang diketahui, dan faktor skala s model.
2. Untuk setiap titik *edge scene* x, dan untuk setiap faktor skala terkuantisasi s yang mungkin dan sudut rotasi β model, *vote* lokasi $\boldsymbol{p}_0$ yang mungkin. Secara lebih rinci, untuk setiap entri $\boldsymbol{r} = (r, \alpha)$ pada indeks $\theta(\boldsymbol{x}) - \beta$, tambah nilai pada sel *array* akumulator
3. Maksima pada *array A* berkorespondensi pada lokasi titik referensi untuk faktor rotasi dan skala model yang mungkin.

Kurva analitik, jenis masalah yang dapat ditangani oleh Hough standar, dapat dimasukkan parameter transformasi sistem koordinat – translasi pada x_0 dan y_0, rotasi β, dan faktor skala *s*. Contohnya persamaan lingkaran diketahui

$$(x - x_0)^2 + (y - y_0)^2 = s^2,$$

dan titik referensi dapat dipilih sebagai pusatnya. Diketahui arah gradien θ pada titik x pada lingkaran dan sebuah faktor skala s, lokasi titik referensi (x_0, y_0) yang mungkin adalah $x + r$ dimana r diketahui dalam bentuk polar yakni

$$|r| = s \text{ dan sudut}(r) = \theta$$

Tabel R tidak perlu dibuat karena vektor referensi pada index θ hanya vektor satuan pada arah gradien θ.

Objek yang dibuat dari bagian-bagian subnya dapat dikenali dengan menggabungkan tabel R setiap bagian sub ke dalam satu tabel R global untuk keseluruhan objek. Pengembangan lainnya dari GHT dasar adalah menggunakan strategi kompleks lainnya untuk menambah nilai pada *array* akumulator, selain menambah nilai dari satu kesatuan objek [11].

Pada intinya algoritma GHT memerlukan tiga parameter Hough: sudut gradient $\emptyset$, jarak relatif ke titik referensi r, dan sudut dasar α untuk menghasilkan tabel referensi atau tabel R terkait seperti yang ditunjukkan pada Tabel 2.1. Sudut gradien

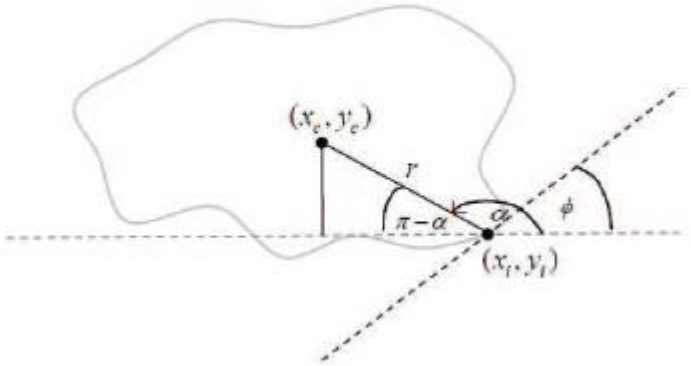
didefinisikan sebagai arah gradien pada setiap titik tepi objek-menarik. Centroid objek adalah titik referensi; karenanya, jarak r hanyalah jarak Euclidian antara titik tepi dan pusat massa. Sudut dasar adalah sudut antara sumbu x dan r, diukur dalam arah berlawanan arah jarum jam seperti yang ditunjukkan pada tabel 2.1.

Tabel 2.1 Tabel R

φ_1	$(r_1, \theta_1), (r_2, \theta_2), (r_3, \theta_3) \dots \dots \dots \dots$
φ_2	$(r_{20}, \theta_{20}), (r_{21}, \theta_{21}), (r_{22}, \theta_{22}) \dots \dots \dots \dots$
…….	……
φ_{360}	$(r_{40}, \theta_{40}), (r_{41}, \theta_{41}), (r_{42}, \theta_{42}) \dots \dots \dots \dots$

Tabel R mewakili bentuk objek yang menarik. Secara teoritis, objek apa pun yang muncul dalam gambar berbeda dengan orientasi dan skala yang sama harus menghasilkan tabel R yang sama. Sebagai hasilnya, pola R-table dapat digunakan untuk klasifikasi objek. Menggunakan koordinat titik tepi yang paling signifikan, GHT menghitung centroid objek, jarak ke centroid, sudut dasar dan arah gradien pada setiap titik tepi sebagai berikut.

Tabel 2.1 menunjukkan format skematik dari tabel referensi yang memiliki vektor perpindahan yang diindeks oleh cekung, jari-jari dan sudut normal. Tabel referensi yang diusulkan dibagi menjadi dua kelompok sesuai dengan tanda-tanda jari-jari. Ingat bahwa jari-jari positif mewakili busur cembung dan jari-jari negatif mewakili busur cekung. Entri dalam setiap kelompok pertama-tama disortir dalam urutan non-menurun oleh jari-jari, dan kemudian diurutkan dalam urutan non-menurun oleh sudut normal untuk setiap jari-jari yang diberikan. Prosedur penyortiran dilakukan secara off-line dalam fase konstruksi tabel referensi. Konkavitas, jari-jari dan sudut normal dari titik batas digunakan sebagai indeks dan batasan untuk menemukan pasangan yang cocok yang benar antara titik-titik objek sensorik dan model. Fitur titik ini menghilangkan alarm palsu pada fase pembuatan suara.



Gambar 2.1 Geometri digunakan untuk membentuk tabel R

Jarak antara *centroid* (x_c, y_c) dan titik tepi ke- i (x_i, y_i) pada bidang koordinat Kartesius dihitung dengan

$$r_i = \sqrt{(x_c - x_i)^2 + (y_c - y_i)^2}$$

Dari gambar 2.1, derajat sudut dasar adalah :

$$\propto = \begin{cases} \tan^{-1}\left(\dfrac{y_c - y_i}{x_c - x_i}\right), & y_c - y_i \geq 0 \\ 360 + \tan^{-1}\left(\dfrac{y_c - y_i}{x_c - x_i}\right), & y_c - y_i \leq 0 \end{cases}$$

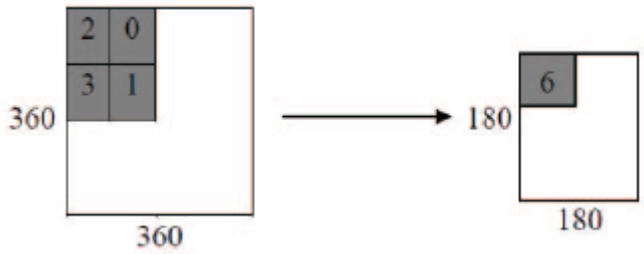
Kemudian sudut gradient dihitung menggunakan

$$\varphi = \tan^{-1}\left(\frac{\Delta y}{\Delta x}\right)$$

Dimana Δx dan Δy adalah berturut-turut selisih arah horizontal dan vertikal.

Tabel R diformat ulang untuk mendapatkan peta fitur-Hough. Fitur Hough kemudian diekstraksi dengan mengidentifikasi puncak signifikan di peta menjadi matriks persegi ukuran 360 × 360 yang nilai baris dan kolomnya masing-masing sesuai dengan sudut gradien φ dan sudut normal α. Nilai-nilai tabel R diurutkan berdasarkan baris untuk semua (r, θ) pasangan yang sama. Jumlah total kejadian θ $k$emudian diformat ke dalam tabel berbeda dengan koordinat (φ, θ). Misalnya, jika θ = 185

muncul 50 kali dalam baris kelima dari tabel R, nilai pada koordinat (5, 185) dari peta diatur menjadi 50.



Gambar 2.2 Remapping saat n = 2

Peta pertama kali dibagi menjadi sub-blok yang lebih kecil dari n x n, di mana n adalah pembagi 360. Setiap sub-blok dari n x n kemudian direduksi menjadi satu elemen yang nilainya diperoleh dengan rata-rata semua elemen yang lebih kecil blok. Akibatnya, peta asli dikurangi atau dipetakan kembali ke N x N, di mana N = 360 / n dengan entri yang merupakan rata-rata dari sub-blok yang sesuai. Proses *remapping* ditunjukkan pada Gambar 3. Koordinat puncak dominan dari peta fitur Hough kemudian diekstraksi untuk membentuk vektor fitur, yang kemudian dimasukkan ke dalam jaringan saraf.

Keberhasilan *hough transform* dilihat dari aspek globalnya, tidak diperlukan pengetahuan prior pada distribusi poin, tetapi proses pemungutan suara (*voting*) setiap poin mengatarkan pada kemunculan puncak akumulator. Prosedur *voting* menjadi kelebihan *hough transform* dalam menyambungkan *edge* yang terputus karena semua titik akan dilakukan *voting* untuk mendapatkan bentuk tertentu [23].

2.4 *Dimensionality Reduction*

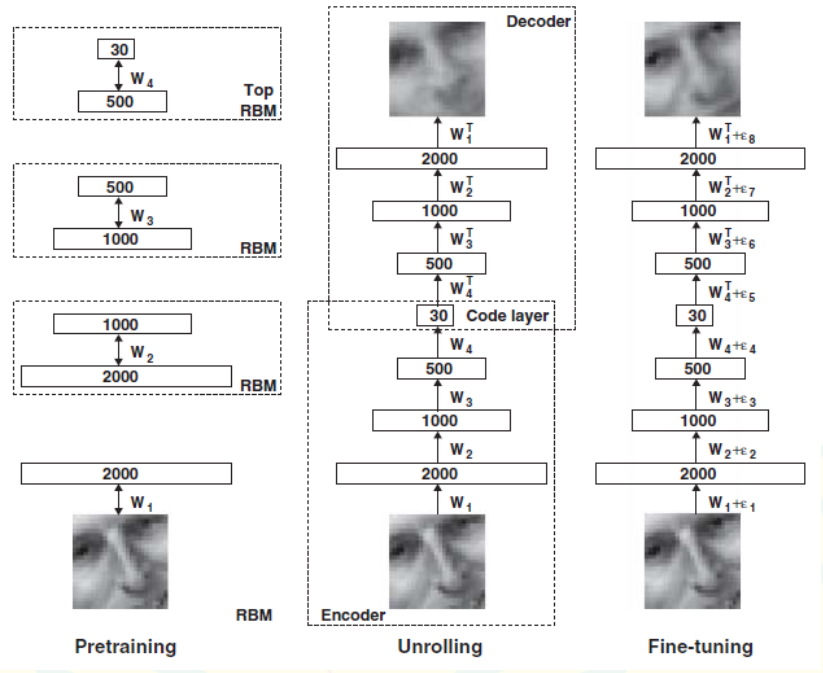
Banyak data yang dapat dilihat sebagai matriks besar. Reduksi dimensi atau *dimensionality reduction* dalam statistik, pembelajaran mesin, dan teori informasi adalah proses mengurangi jumlah variabel acak yang dipertimbangkan dengan memperoleh seperangkat variabel utama. Pendekatan dibagi menjadi pemilihan fitur dan ekstraksi fitur [24].

Ekstraksi fitur mengubah data dalam ruang dimensi tinggi menjadi ruang dimensi yang lebih sedikit. Transformasi data bisa dalam bentuk linear, seperti dalam analisis komponen utama (PCA) dan nonlinear. Untuk data multidimensi, representasi tensor dapat digunakan dalam pengurangan dimensi melalui pembelajaran ruang bagian multilinear. Selain PCA, adapun metode ekstraksi fitur lainnya antara lain : vektor eigen dan nilai eigen, dekomposisi nilai tunggal (SVD), dekomposisi CUR [15].

Pengurangan dimensi atau *dimensionality reduction* memfasilitasi klasifikasi, visualisasi, komunikasi, dan penyimpanan data dimensi tinggi. Metode sederhana dan banyak digunakan adalah analisis komponen utama (PCA), yang menemukan arah varians terbesar dalam kumpulan data dan mewakili setiap titik data dengan koordinatnya di sepanjang masing-masing arah ini. Generalisasi nonlinear PCA yang menggunakan jaringan encoder multilayer adaptif ditujukan untuk mengubah data dimensi tinggi menjadi kode dimensi rendah dan jaringan dekoder serupa untuk memulihkan data dari kode.

2.5 *Autoencoder*

Cara kerjanya dimulai dengan bobot acak di dua jaringan dilatih bersama dengan meminimalkan perbedaan antara data asli dan rekonstruksinya. Gradien yang diperlukan mudah diperoleh dengan menggunakan aturan rantai untuk melakukan *backpropagation error derivative* terlebih dahulu melalui jaringan decoder dan kemudian melalui jaringan encoder. Keseluruhan sistem ini dinamakan *autoencoder* seperti pada gambar berikut.



Gambar 2.3 Autoencoder

Adapun tujuan *pretraining* adalah untuk mengatasi beberapa masalah berikut : besar gradien pada lapisan yang lebih rendah dan lapisan yang lebih tinggi berbeda, *stochastic gradient descent* sebagai fungsi objektif sulit mendapatkan nilai optimum lokal, dan *deep network* memiliki banyak parameter yang mana tidak dapat melakukan generalisasi dengan baik [14].

Prosedur *pretraining* untuk data biner diperkenalkan untuk mengoptimasi bobot pada *autoencoder* non-linear yang memiliki banyak lapisan tersembunyi. Ensembel vektor biner (misal Gambar) dapat dimodelkan menggunakan jaringan dua *layer* yang dinamakan *Restricted Boltzmann Machine* (RBM) (5, 6) di mana stokastik, piksel biner terhubung ke stokastik, detektor fitur biner menggunakan koneksi berbobot simetris. Pixel berhubungan dengan B unit yang terlihat dari RBM karena *state*-nya diamati; detektor fitur bersesuaian dengan B unit yang disembunyikan. Konfigurasi gabungan $(\boldsymbol{v}, \boldsymbol{h})$ dari unit tampak dan tersembunyi memiliki energi yang diberikan oleh

$$E(\boldsymbol{v}, \boldsymbol{h}) = -\sum_{\boldsymbol{i} \in \boldsymbol{pixels}} \boldsymbol{b_i v_i} - \sum_{\boldsymbol{j} \in \boldsymbol{features}} \boldsymbol{b_j h_j} - \sum_{\boldsymbol{i,j}} \boldsymbol{v_i h_j w_{ij}}$$

di mana $v_i$dan h_j adalah *state* biner dari pixel i dan fitur j, b_i dan b_j adalah bias mereka, dan w_{ij} adalah bobot di antara mereka. Jaringan menetapkan probabilitas untuk setiap gambar yang mungkin melalui fungsi energi ini. Probabilitas gambar pelatihan dapat ditingkatkan dengan mengatur bobot dan bias untuk mengurangi energi gambar tersebut dan meningkatkan energi yang serupa, sehingga menyerupai data sebenarnya. Sebagai contoh gambar *training*, *state* biner h_j setiap pendeteksi fitur j diatur menjadi 1 dengan probabilitas $\sigma(b_j + \sum_i v_i w_{ij})$, dimana $\sigma(x)$ merupakan fungsi logistic $1/[1 + \exp(-x)]$, $b_j$adalah bias j, v_i adalah *state* piksel i, dan w_{ij}adalah bobot antara i dan j. Setelah *state* biner dipilih untuk satuan tersembunyi, sebuah *confabulation* dihasilkan dengan cara mengatur setiap v_i menjadi 1 dengan probabilitas $\sigma(b_i + \sum_j h_j w_{ij})$, dimana b_i adalah bias i. *State* pada satuan tersembunyi diperbaharui sekali lagi sehingga dapat merepresentasikan fitur *confabulation*. Adapun selisih bobot sebagai berikut.

$$\Delta w_{ij} = \varepsilon((v_i h_j)_{data} - (v_i h_j)_{recon})$$

Dimana ε merupakan *learning rate*, $(v_i h_j)_{data}$merupakan pecahan waktu dengan piksel i dan pendeteksi fitur j bersama ketika detektor fitur digerakkan oleh data, dan $(v_i h_j)_{recon})$ adalah pecahan yang sesuai untuk *confabulation*. Versi sederhana dari aturan belajar yang sama digunakan untuk bias. Pembelajaran bekerja dengan baik meskipun tidak persis mengikuti gradien probabilitas log dari data pelatihan.

Satu lapisan fitur biner bukan cara terbaik untuk memodelkan struktur dalam satu set gambar. Setelah mempelajari satu lapisan pendeteksi fitur, data digunakan untuk mempelajari lapisan kedua fitur. Lapisan pertama fitur detektor kemudian menjadi unit yang terlihat untuk mempelajari RBM berikutnya. Pembelajaran lapisan demi lapisan ini dapat diulang sebanyak-banyaknya. Dapat ditunjukkan bahwa menambahkan lapisan tambahan selalu meningkatkan batas bawah pada probabilitas log yang diberikan oleh model pada data pelatihan, asalkan jumlah detektor fitur per

lapisan tidak berkurang dan bobotnya diinisialisasi dengan benar. Batas ini tidak berlaku ketika lapisan yang lebih tinggi memiliki lebih sedikit fitur pendeteksi, tetapi algoritma pembelajaran lapis demi lapis tetap merupakan cara yang sangat efektif untuk menguji berat bobot dari *deep autoencoder*. Setiap lapisan fitur menangkap korelasi tingkat tinggi yang kuat antara aktivitas unit di lapisan di bawah ini. Untuk berbagai set data, ini adalah cara yang efisien untuk secara progresif mengungkapkan struktur nonlinear dimensi rendah.

Setelah *pretraining* beberapa lapisan detektor fitur, model dibuka [(Gbr. 1) untuk menghasilkan jaringan encoder dan decoder yang awalnya menggunakan bobot yang sama. Tahap finetuning global kemudian menggantikan kejadian stokastik dengan deterministik, probabilitas bernilai nyata dan menggunakan *backpropagation* melalui seluruh autoencoder untuk menyempurnakan bobot untuk rekonstruksi optimal.

Untuk data kontinu, unit tersembunyi dari RBM tingkat pertama tetap biner, tetapi unit yang terlihat digantikan oleh unit linear dengan *noise* Gaussian (10). Jika *noise* ini memiliki varian unit, aturan pembaruan stokastik untuk unit tersembunyi tetap sama dan aturan pembaruan untuk unit terlihat i adalah sampel dari Gaussian dengan varian unit dan rata-rata $b_i + \sum_j h_j w_{ij}$.

Tanpa *pretraining*, *autoencoder* yang sangat dalam selalu merekonstruksi rata-rata data pelatihan, bahkan setelah fine-tuning yang berkepanjangan. *Autoencoder* yang lebih tipis dengan satu lapisan tersembunyi antara data dan kode dapat dipelajari tanpa *pretraining*, tetapi *pretraining* sangat mengurangi total waktu pelatihan. Ketika jumlah parameter sama, *autoencoder* yang dalam dapat menghasilkan kesalahan rekonstruksi yang lebih rendah pada data uji daripada yang dangkal, tetapi tidak ketika jumlah parameter meningkat.

Proses *pretraining* lapis demi lapis dapat digunakan untuk klasifikasi. Tingkat kesalahan terbaik yang dilaporkan adalah 1,4% untuk *support vector machine* (SVM). Setelah *pretraining* lapis demi lapis dalam jaringan 784-500-500-2000-10,

backpropagation menggunakan keturunan yang paling curam dan tingkat pembelajaran yang kecil mencapai 1,2%. *Pretraining* membantu generalisasi karena memastikan bahwa sebagian besar informasi dalam bobot berasal dari pemodelan gambar. Informasi yang sangat terbatas dalam label hanya digunakan untuk sedikit menyesuaikan bobot yang ditemukan dengan *pretraining* [12].

Hasil eksperimen pada klasifikasi digit MNIST menggunakan lima algoritma : *deep net* menggunakan *unsupervised pretraining*, *deep net* menggunakan *auto-associator pretraining*, *deep net* menggunakan *supervised pretraining*, *deep net* tanpa *pretraining* dan *shallow net* tanpa *pretraining* menunjukkan bahwa tanpa *pretraining*, *deep network* tidak menghasilkan performa yang lebih baik daripada *shallow network* dan didapatkan hasil yang sama ketika menerapkan *autoencoder* daripada *Restricted Boltzmann Machine* (RBM) [3].

Sebuah *autoencoder* memetakan vektor masukan $x \in [0,1]^d$ ke dalam representasi tersembunyi $y \in [0,1]^{d'}$ melalui pemetaan deterministik $y = f_0(x) = s(\mathbf{W}\mathbf{x} + \mathbf{b})$, dengan parameter $\theta = \{\mathbf{W}, \mathbf{b}\}$. $\mathbf{W}$ adalah matriks bobot d'x d dan $\mathbf{b}$ adalah vektor bias. Hasil representasi laten $\boldsymbol{y}$ kemudian dipetakan kembali ke sebuah vektor yang direkonstruksi $z \in [0,1]^d$ dalam ruang masukan $\mathbf{z} = g_{\theta'}(\mathrm{y}) = s(\mathbf{W}'\mathbf{y} + \mathbf{b}')$ dengan $= \{\mathbf{W}', \mathbf{b}'\}$. Matriks bobot $\mathbf{W}'$ dari pemetaan terbalik bisa secara opsional dibatasi oleh $\mathbf{W}' = \mathbf{W}^{\mathrm{T}}$, sehingga *autoencoder* memiliki bobot yang terikat. Setiap *training* $\mathbf{x}^{(i)}$ dipetakan ke $\mathbf{y}^{(i)}$ yang bersesuaian dan dilakukan rekonstruksi $\mathbf{z}^{(i)}$. Parameter model ini dioptimasi untuk meminimalisasi *average reconstruction error* :

$$\theta^*, \theta'^* = \arg\min_{\theta,\theta'} \frac{1}{n} \sum_{i=1}^{n} L\,(x^{(i)}, z^{(i)})$$

$$= \arg\min_{\theta,\theta'} \frac{1}{n} \sum_{i=1}^{n} L\,(x^{(i)}, g_{\theta'}\left(f_0\left(x^{(i)}\right)\right))$$

Dimana L adalah *loss function* seperti *squared error* $L(\mathbf{x}, \mathbf{z}) = \left|\left|\mathbf{x} - \mathbf{z}\right|\right|^2$. Sebuah *loss* alternative yang diambil dari interpretasi $\mathbf{x}$ dan $\mathbf{z}$ sebagai vektor-vektor bit

dan vektor-vektor probabilitas bit (Bernoullis) merupakan *reconstruction cross-entropy*.

$$L_{\mathrm{H}}(\mathrm{x},\mathrm{z}) = \mathrm{H}(B_x||B_z)$$

$$= -\sum_{k=1}^{d}[x_k \log z_k + (1-x_k)\log(1-z_k)]$$

Perlu dicatat bahwa jika **x** adalah vektor biner, $L_{\mathrm{H}}(\mathrm{x},\mathrm{z})$ adalah kemungkinan-log negatif untuk contoh **x**, dengan parameter $\boldsymbol{z}$ diketahui. Persamaan 1 dengan $L = L_{\mathrm{H}}$ dapat ditulis menjadi

$$\theta^*, \theta'^* = \arg\min_{\theta,\theta'} \mathrm{E}_{\mathrm{q}^0(X)}[L_{\mathrm{H}}\left(X, g_{\theta'}\left(f_\theta(X)\right)\right)]$$

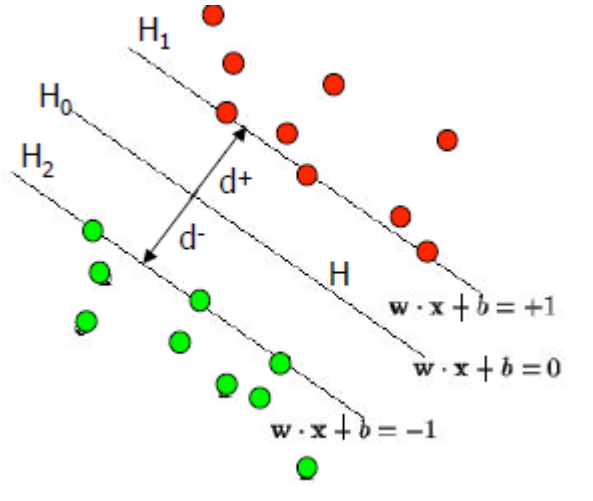
Dimana $q^0(X)$ menunjukkan distribusi empiris yang diasosiasikan ke masukan *training* n. Optimasi ini akan diteruskan oleh *stochastic gradient descent* [31].

2.6 *Neural Network*

Jaringan saraf atau *neural network* adalah kumpulan elemen, unit atau node pemrosesan yang saling berhubungan, yang fungsinya secara longgar didasarkan pada neuron hewan. Kemampuan pemrosesan jaringan disimpan dalam kekuatan interunitconnection, atau bobot, yang diperoleh dengan proses adaptasi, atau belajar dari, serangkaian pola pelatihan [9].

2.7 *Support Vector Machine* (SVM)

Support vector machine (SVM) adalah metode *supervised learning* yang menghasilkan fungsi pemetaan masukan-keluaran dari satu set data pelatihan berlabel. Fungsi pemetaan dapat berupa fungsi klasifikasi, yaitu kategori dari data input, atau regresi fungsi. Untuk klasifikasi, fungsi kernel nonlinier sering digunakan untuk mengubah data input ke ruang fitur dimensi tinggi di mana data input menjadi lebih dapat dipisahkan dibandingkan dengan ruang input asli. Hyperplanes margin maksimum kemudian dibuat. Model yang dihasilkan tergantung pada hanya sebagian dari data pelatihan di dekat batas kelas. Demikian pula, model yang dihasilkan oleh *Support Vector Regression* mengabaikan data pelatihan apa pun yang cukup dekat dengan prediksi model. MV juga dikatakan milik "metode kernel". Selain landasan matematika yang kuat dalam teori pembelajaran statistik, SVM telah menunjukkan kinerja yang sangat kompetitif dalam berbagai aplikasi dunia nyata, seperti bioinformatika, penambangan teks, pengenalan wajah, dan pemrosesan gambar, yang telah menetapkan SVM sebagai salah satu alat canggih untuk pembelajaran mesin dan penambangan data, serta softcomputing lainnya teknik, misalnya, jaringan saraf dan sistem fuzzy [33].



Gambar 2.4 Hyperplane

Prinsip dasar di balik SVM adalah menggambar *hyperplane* yang memisahkan dua kelas. Yang terdekat pada titik data ke *hyperplane* disebut vektor dukungan. *Hyperplane* digambar berdasarkan vektor-vektor dukungan ini. Biasanya *hyperplane*, yang memiliki jarak maksimum dari vektor dukungan, adalah *hyperplane* optimal dan jarak antara *hyperplane* dan vektor dukungan dikenal sebagai margin. Rumus matematis sederhana SVM linear yakni dalam bentuk :

$$f(x) = w \cdot x + b$$

Formulasi paling umum SVM menemukan *hyperplane* dalam ruang yang berbeda dari data input x dalam ruang fitur yang diinduksi oleh kernel K (kernel mendefinisikan produk titik di ruang itu (Wahba, 1990)). Melalui kernel K, ruang hipotesis didefinisikan sebagai seperangkat "*hyperplanes*" dalam ruang fitur yang diinduksi oleh K. Selain itu, hal ini dapat dilihat sebagai seperangkat fungsi dalam Ruang Reproducing Kernel Hilbert Space (RKHS) yang ditentukan oleh K (Wahba, 1990), (Vapnik, 1998). Singkatnya, ruang hipotesis yang digunakan oleh SVM adalah himpunan bagian dari *hyperplanes* yang didefinisikan dalam beberapa ruang - sebuah RKHS. Ruang ini dapat ditulis secara formal sebagai berikut.

$$\{f : ||f||_K^2 < \infty\}$$

K adalah kernel yang mendefinisikan RKHS, dan $||f||_K^2$ adalah norma fungsi RKHS (Wahba, 1990). Sebagai contoh, untuk kasus linear yang disebutkan di atas, K adalah kernel K (x1, x2) = x1·x2, fungsi yang dipertimbangkan adalah dari bentuk f (x) = w·x + b, dan norma RKHS dari fungsi-fungsi ini hanyalah norma w, yaitu $||f||_K^2 = ||w||^2$}. Tujuan SVM adalah untuk menemukan solusi dengan norma RKHS "optimal", yaitu untuk menemukan $||w||$ yang optimal.

SVM adalah mesin pembelajaran yang meminimalkan kesalahan empiris sambil memperhitungkan "kompleksitas" ruang hipotesis yang digunakan dengan juga meminimalkan norma RKHS dari solusi $||f||_K^2$. SVM dalam praktiknya meminimalkan pertukaran antara kesalahan empiris dan kompleksitas ruang hipotesis. Secara formal ini dilakukan dengan memecahkan masalah minimisasi berikut:

Klasifikasi SVM

$$\min_f ||f||_K^2 + C \sum_{i=1}^{1} |1 - y_i f(x_i)|_+$$

Regresi SVM

$$\min_f ||f||_K^2 + C \sum_{i=1}^{1} |y_i - f(x_i)|_\varepsilon$$

C adalah "parameter regularisasi" yang mengontrol pertukaran antara kesalahan empiris dan kompleksitas ruang hipotesis yang digunakan [34].