PENGARUH JUMLAH VIDEO LATIH UNTUK SETIAP KELAS PADA APLIKASI *NEAR DUPLICATE VIDEO RETRIEVAL* MENGGUNAKAN METODE *T-DISTRIBUTED UNSUPERVISED STOCHASTIC MULTI-VIEW HASHING*

Ruben Stefanus, Riko Arlando Saragih

Fakultas Teknik, Program Studi Teknik Elektro Universitas Kristen Maranatha

Jl. Surya Sumantri No.65, Bandung 40164, Indonesia

rstefanus66@gmail.com

riko_saragih@yahoo.com

Abstrak — **Aplikasi NDVR muncul akibat banyaknya video yang hampir duplikat dari video asli. Video yang hampir duplikat ini dapat dimanfaatkan dan dikomersialkan tanpa sepengetahuan dari pemilik hak cipta. Pada sistem NDVR terdapat masalah ketidakseimbangan jumlah video latih untuk setiap kelas. Oleh karena itu perlu dilakukan pengujian sistem terhadap jumlah video latih untuk setiap kelas yang berbeda untuk melihat pengaruhnya. Pada penelitian ini, perancangan sistem NDVR menggunakan metode t-USMVH. Sistem NDVR ini terbagi menjadi 4 tahapan yaitu ekstraksi *keyframe*, ekstraksi ciri, pelatihan sistem, dan juga pencocokan kode *hash* antarvideo. Pada ekstraksi *keyframe* proses pemilihan *keyframe* berdasarkan *sampling* per detik namun tetap mempertimbangkan perubahan citra antardetik. Pada ekstraksi ciri didapatkan ciri lokal dan ciri global, yaitu *Hue Saturation Value* (HSV) dan *Local Directional Pattern* (LDP). Untuk pelatihan sistem berdasarkan arsitektur *backpropagation*, dan pencocokan kode *hash* antarvideo menggunakan metode *Hamming Distance*. Pengujian dilakukan terhadap kombinasi variasi ciri, variasi jumlah iterasi, dan variasi jumlah video latih untuk setiap kelas. Terdapat 2 variasi jumlah video latih untuk setiap kelas yaitu, *imbalance* query video dan *balance* query video. Dari hasil nilai MAP yang didapatkan terlihat bahwa nilai MAP untuk *balance* query video lebih besar dibandingkan dengan *imbalance* query video pada semua variasi iterasi dan semua variasi ciri. Sedangkan untuk sistem *balance* query video menghasilkan nilai MAP yang lebih baik jika menggunakan gabungan ekstraksi ciri lokal dan global dibandingkan denganhanya satu ekstraksi ciri.**

Kata kunci— *Keyframe*, NDVR, t-USMVH, HSV, LDP, *balance* query video, *imbalance* query video, MAP.

Abstract — ***The NDVR application appear as a result the large number of near duplicate videos from the original video. This near duplicate video can be used and commercialized without the knowledge of the copyright owner. In the NDVR system there is a problem of the imbalance in the number of training videos for each class. Therefore it is necessary to test the system for the number of videos for each different class to see the effect.In this Final Project, the design of the NDVR system use the t-USMVH method. The NDVR system is divided into 4 stages namely keyframe extraction, feature extraction, system training, and also matching intervideo hash codes. In keyframe extraction the process of selecting keyframes is based on sampling per second but still consider intersecond image changes. In feature extraction, local feature and alobal feature are obtained, namely Hue Saturation Value (HSV) and Local Directional Pattern (LDP). For training systems based on backpropagation architecture, and matching intervideo hash codes using the Hamming Distance method. Experiments were performed on a combination of variations in feature, variations in the number of iterations, and variations in the number of training videos for each class. There are 2 types of variations in the number of training videos for each class, namely, video query imbalance and video query balance. From the results of the MAP values obtained, it can be seen that the MAP value for video query balance is greater than video query imbalance on all variation of iteration and all variation of feature. While for the video query balance produce a better MAP value if using combination of local and global feature extraction compared to only one feature extraction.***

Keywords— *Keyframe*, NDVR, t-USMVH, HSV, LDP, *balance* query video, *imbalance* query video, MAP.

I. PENDAHULUAN

Perkembangan teknologi internet terus berkembang dan popularitas produk digital, aktivitas *online* yang berkaitan dengan video seperti pengunduhan, pengunggahan, dan menonton video secara *online* mengalami peningkatan yang signifikan dalam beberapa tahun terakhir. Hal ini menyebabkan jumlah video pada internet semakin banyak dan meningkatkan permintaan untuk kebutuhan klasifikasi video, pengenalan objek, deteksi orisinalitas video, dan lain-lain[1], [2].

Akibat meningkatnya kebutuhan video dalam dunia maya, maka muncul masalah video-video yang hampir duplikat dari video asli. Video yang hampir duplikat dari video asli ini merupakan video asli yang telah dilakukan perubahan pada frame tertentu.

Video yang hampir duplikat dari video asli ini disebut sebagai *Near Duplicate Video* (NDV). Pemilik hak cipta dari video asli tersebut akan mengalami kerugian akibat adanya video yang hampir duplikat ini karena video tersebut dapat digunakan dan dikomersialkan tanpa sepengetahuan dari pemilik hak cipta.

Oleh karena itu, dilakukanlah *Near Duplicate Video Retrieval* (NDVR). NDVR merupakan proses pencarian video yang hampir duplikat dari video asli. Proses NDVR secara umum diawali dengan ekstraksi *keyframe* dari video. *Keyframe* adalah frame citra yang dapat mewakili konten dan informasi yang penting dari suatu video[4]. Kemudian setiap *keyframe* dilakukan ekstraksi ciri dan terakhir dilakukan pencarian kemiripan antara video menggunakan informasi ciri tersebut.

Penelitian terkini tentang NDVR, khususnya *large-scale near-duplicate video retrieval*, dilakukan oleh Yanbin Hao dan kawan-kawan[1]. Dalam penelitian mereka diusulkan sebuah algoritma yang berakurasi tinggi, efisien, memiliki skalabilitas yang baik, dan menggunakan tidak hanya sebuah vektor ciri tunggal (*single feature vector*) untuk menemukan *near duplicate video*. Skalabilitas merupakan kemampuan algoritma menangani komputasi untuk jumlah video yang besar.

Algoritma yang digunakan adalah *Stochastic Multi-View Hashing* (SMVH) yang menggunakan beberapa jenis informasi dari video, seperti ciri global, ciri lokal, dan *keyframe*. Penggunaan beberapa jenis informasi dari video dapat meningkatkan kinerja NDVR namun mengakibatkan data menjadi berdimensi tinggi. Karena data yang berdimensi tinggi dapat mengakibatkan waktu komputasi yang lama dan kompleksitas yang tinggi, maka dilakukanlah reduksi dimensi data. Reduksi dimensi data berguna untuk mengurangi waktu komputasi dan kompleksitas.

SMVH menerapkan teknik *hashing* yang memungkinkan pencarian kemiripan antara video secara cepat. *Hash* adalah suatu metode untuk mengenkode objek seperti video, gambar, atau dokumen menjadi deretan biner yang memiliki ukuran tetap melalui proses pemetaan[2]. Metode *hash* ini mempunyai kelebihan dalam pencarian *near duplicate video* karena waktu komputasi yang cepat dan mengurangi penggunaan memori saat komputasi.

Pada sistem NDVR juga terdapat masalah ketidakseimbangan jumlah video latih (*imbalance data*) yang dapat mempengaruhi kinerja sistem NDVR[5]. Ketidakseimbangan jumlah video latih merupakan suatu keadaan dimana distribusi kelas video tidak seimbang, jumlah kelas video yang satu lebih sedikit atau lebih banyak dibandingkan dengan jumlah kelas video lainnya. Kelas video yang lebih sedikit disebut kelas minoritas, sedangkan kelas video yang lebih banyak disebut kelas mayoritas. Dalam pengklasifikasian video biasanya akurasi untuk kelas mayoritas jauh lebih tinggi daripada kelas minoritas. Untuk kasus NDVR biasanya *near duplicate video* jumlahnya lebih sedikit (kelas minoritas) dibandingkan jumlah video yang berbeda (kelas mayoritas). Namun, informasi *near duplicate video* sangat penting untuk berhasil diidentifikasi.

Dalam penelitian ini akan dibuat sistem NDVR dengan metode berdasarkan pendekatan dari Yanbin Hao dan kawan-kawan[2], lalu akan dievaluasi pengaruh jumlah video latih untuk setiap kelas pada kinerja sistem. Pada penelitian ini jenis deskriptor lokal yang akan digunakan adalah *Local Directional Pattern* (LDP), bukan menggunakan *Local Binary Pattern* (LBP) seperti pada metode yang diusulkan Yanbin Hao dan kawan-kawan[2]. Alasan pemilihan deskriptor lokal LDP karena deskriptor lokal LBP tidak dapat mengakomodir keberadaan *noise* dan variasi iluminasi yang non-monotonik[6]. Dan pada bagian ekstraksi *keyframe* akan digunakan metode berdasarkan perbedaan tepi dan bentuk poligon dari citra.

II. PENELITIAN SEBELUMNYA

Misalnya terdapat sejumlah V video, lalu didapatkan sejumlah *keyframe* yang diekstraksi dari setiap video menggunakan metode *uniform time sampling*. Asumsikan sejumlah n *keyframe* diekstraksi dari V video. Untuk setiap *keyframe*, diekstraksi beberapa ciri untuk mencirikan informasi dari *keyframe* tersebut. Kemudian informasi dari setiap ciri menjadi masukan untuk metode t-USMVH.

A. FUNGSI *HASH*[2]

Fungsi *hash* adalah fungsi matematis yang mengubah nilai masukan yang memiliki ukuran tertentu menjadi suatu nilai yang memiliki ukuran yang tetap[13]. Berdasarkan ide ini maka fungsi *hash* mulai digunakan untuk memetakan data yang berukuran besar, seperti gambar, video atau dokumen digital lainnya menjadi kode biner dengan panjang bit tertentu[2].

$$\boldsymbol{z}_{il} = sigmoid\left(\sum_{g=1}^{m}\sum_{j=1}^{d_g} \boldsymbol{x}_{ij}^{(g)} \boldsymbol{w}_{jl}^{(g)} + \boldsymbol{b}_l\right) \quad (1)$$

Keterangan :
$\boldsymbol{z}_{il}$: kode *hash*
$sigmoid$: fungsi aktivasi
m : tipe ciri (HSV, LDP)
d_g : dimensi dari histogram ciri
$\boldsymbol{x}_{ij}^{(g)}$: matriks histogram ciri dari *keyframe*
$\boldsymbol{w}_{lj}^{(g)}$: matriks nilai bobot
$\boldsymbol{b}_l$: matriks nilai bias
i, j : indeks *keyframe*
l : panjang kode *hash*

B. PELUANG *MULTI-VIEW DATA* ($\boldsymbol{P}$)[2][14]

Multi-view data adalah data yang memiliki banyak informasi ciri. Peluang *multi-view data* ($\boldsymbol{P}$) disebut sebagai peluang aktual. Tahapan perhitungan peluang untuk *multi-view data* ($\boldsymbol{P}$) adalah sebagai berikut :

1. Hitung matriks peluang kemiripan antara ciri ($\boldsymbol{p}$). Matriks ini mengukur seberapa dekat nilai x_j dari x_i dengan memperhitungkan distribusi *Gaussian* menggunakan Persamaan (2).

$$\boldsymbol{p}_{j|i}^{(g)} = \frac{\exp\left(-\frac{\left\|x_i^{(g)} - x_j^{(g)}\right\|_2^2}{2\sigma_{ig}^2}\right)}{\sum_{l \neq i} \exp\left(-\frac{\left\|x_i^{(g)} - x_l^{(g)}\right\|_2^2}{2\sigma_{ig}^2}\right)} \quad (2)$$

2. Hitung matriks koefisien *Jaccard* ($\boldsymbol{J}$). Koefisien *Jaccard* berguna untuk menghitung seberapa besar kemiripan antara dua ciri *keyframe* menggunakan Persamaan (3).

$$\boldsymbol{J}_{ij}^{(g)} = \frac{\left|x_i^{(g)} \cap x_j^{(g)}\right|}{\left|x_i^{(g)} \cup x_j^{(g)}\right|} \quad (3)$$

3. Hitung matriks total kemiripan ($\boldsymbol{P_c}$) dengan persamaan (4).

$$\boldsymbol{P_c} = \frac{1}{2n} \sum_{g=1}^{m} \boldsymbol{J}_{ij}^{(g)} \left(\boldsymbol{p}_{j|i}^{(g)} + \boldsymbol{p}_{i|j}^{(g)}\right) \quad (4)$$

4. Hitung matriks *with-in* video $\left(\boldsymbol{P_w} = p_{ij}^{(w)}\right)$ dengan Persamaan (5). Matriks *with-in* video ini memberikan informasi tentang *keyframe* yang berasal dari suatu video yang sama.

$$p_{ij}^{(w)} = \begin{cases} 1, \ jika \ x_i \ dan \ x_j \ (i \neq j) \ diekstraksi \\ \qquad dari \ video \ yang \ sama \\ 0, \qquad\qquad dalam \ keadaan \ lain \end{cases} \quad (5)$$

5. Hitung matriks peluang *multi-view* (**P**) dengan Persamaan (6).

$$\boldsymbol{P} = \boldsymbol{0.7}\ \boldsymbol{N}(\boldsymbol{P_c}) + \boldsymbol{0.3}\ \boldsymbol{N}(\boldsymbol{P_w}) \quad (6)$$

Keterangan :
i, j : indeks *keyframe*
m : tipe ciri (HSV, LDP)
σ_i^2 : varians dari *keyframe* dengan indeks i
n : jumlah ciri yang digunakan
$\boldsymbol{p}_{i|j}^{(g)}$: transpose dari $\boldsymbol{p}_{j|i}^{(g)}$
$|\,.\,|$: kardinalitas
$\|\,.\,\|_2$: *Euclidian Norm*
$\boldsymbol{z}_i$: kode *hash* untuk *keyframe* dengan indeks i

C. PELUANG *SINGLE-VIEW DATA* ($\boldsymbol{Q}$)[2][14]

Peluang *single-view data* ($\boldsymbol{Q}$) disebut sebagai peluang yang diprediksi, karena setiap kali iterasi nilainya akan berubah. Matriks peluang $\left(\boldsymbol{Q} = \boldsymbol{q}_{ij}\right)$ dibuat berdasarkan kode *hash* yang didapatkan dari fungsi *hash*. Peluang $\left(\boldsymbol{Q} = \boldsymbol{q}_{ij}\right)$ dapat dilihat pada Persamaan (7).

$$\boldsymbol{q}_{ij} = \frac{\left(1 + \left\|z_i - z_j\right\|_2^2\right)^{-1}}{\sum_{k \neq l} \left(1 + \left\|z_k - z_l\right\|_2^2\right)^{-1}} \quad (7)$$

D. PERBEDAAN *KULLBACK-LEIBLER*[2]

Perbedaan *Kullback Leibler* adalah ukuran seberapa besar suatu distribusi peluang berbeda dari distribusi peluang pembandingnya. Perbedaan *Kullback Leibler* adalah salah satu jenis *cost function*. *Cost Function* adalah fungsi yang berguna untuk mengukur seberapa besar kesalahan dalam memodelkan $\boldsymbol{Q}$ dari $\boldsymbol{P}$. Karena *Kullback Leibler* tidak simetris maka dihitunglah fungsi objektif berdasarkan KL_1 dan KL_2 dengan persamaan sebagai berikut :

$$KL_1 = \sum_{t \neq i} p_{it} \ log \frac{p_{it}}{q_{it}} \quad (8)$$

$$KL_2 = \sum_{t \neq i} q_{it} \ log \frac{q_{it}}{p_{it}} \quad (9)$$

$$O = \lambda \ KL_1 + (1 - \lambda) \ KL_2 \quad (10)$$

Keterangan :

KL_1 : *Kullback Leibler 1*
KL_2 : *Kullback Leibler 2*
i, t : indeks *keyframe*
p_{it} : matriks peluang *multi-view data*
q_{it} : matriks peluang *single-view data*
O : fungsi objektif
λ : parameter *balancing* antara KL_1 dan KL_2

E. *GRADIENT DESCENT*[2]

Gradient Descent adalah algoritma pengoptimalan yang mencoba menemukan nilai minimum dari suatu fungsi pada setiap iterasi. *Gradient Descent* adalah metode iterasi yang berarti prosesnya dilakukan secara berulang sampai sejumlah iterasi yang ditetapkan atau sampai mencapai suatu kondisi tertentu. Pada penelitian ini, *Gradient Descent* digunakan untuk meminimalkan perbedaan *Kullback Leibler* terhadap parameter model bobot ($\boldsymbol{w}$) dan bias ($\boldsymbol{b}$).

Tahapan proses *Gradient Descent* adalah sebagai berikut :

1. Inisialisasi nilai bobot ($\boldsymbol{w}$) dan bias ($\boldsymbol{b}$) secara acak diawal
2. Hitung nilai gradien untuk fungsi objektif terhadap bobot dan terhadap bias menggunakan Persamaan (11) dan (12)

$$\frac{\partial O}{\partial w} = \left[\lambda \frac{\partial KL_1}{\partial z} + (1 - \lambda) \frac{\partial KL_2}{\partial z}\right] \frac{\partial z}{\partial w} \quad (11)$$

$$\frac{\partial O}{\partial b} = \left[\lambda \frac{\partial KL_1}{\partial z} + (1 - \lambda) \frac{\partial KL_2}{\partial z}\right] \frac{\partial z}{\partial b} \quad (12)$$

$$\frac{\partial KL_1}{\partial z} = 4 \sum_t (p_{it} - q_{it}) \left(1 + \|z_i - z_t\|_2^2\right)^{-1} \ (z_i - z_t) \quad (13)$$

$\frac{\partial KL_2}{\partial z} = 4\sum_t q_{it}\left(\sum_{k\neq l} q_{kl}\, log\, \frac{q_{kl}}{p_{kl}} - log\, \frac{q_{it}}{p_{it}}\right)(z_i - z_t)\,(1+\|z_i - z_t\|_2^2)^{-1}$ (14)

$$\frac{\partial z}{\partial w} = z\,[1-z]\;x_{ij}^{(g)} \quad (15)$$

$$\frac{\partial z}{\partial b} = z\,[1-z] \quad (16)$$

Keterangan :
$\frac{\partial O}{\partial w}$: nilai gradien fungsi objektif terhadap parameter bobot
$\frac{\partial O}{\partial b}$: nilai gradien fungsi objektif terhadap parameter bias
$\frac{\partial KL_1}{\partial z}$: nilai gradien *Kullback Leibler 1* terhadap kode hash
$\frac{\partial KL_2}{\partial z}$: nilai gradien *Kullback Leibler 2* terhadap kode hash
$\frac{\partial z}{\partial w}$: nilai gradien kode hash terhadap parameter bobot
$\frac{\partial z}{\partial b}$: nilai gradien kode hash terhadap parameter bias
λ : parameter *balancing* antara KL_1 dan KL_2

3. Perbarui nilai bobot $(\boldsymbol{w})$ dan bias $(\boldsymbol{b})$ dengan Persamaan (17) dan (18)

$$w(t+1) = w(t) + \eta\frac{\partial O}{\partial w} + \xi(t)\big(w(t) - w(t-1)\big) \quad (17)$$
$$b(t+1) = b(t) + \eta\frac{\partial O}{\partial b} + \xi(t)\big(b(t) - b(t-1)\big) \quad (18)$$

Keterangan :
$w(t+1)$: nilai bobot terbaru
$b(t+1)$: nilai bias terbaru
$w(t)$: nilai bobot saat ini
$b(t)$: nilai bias saat ini
$w(t-1)$: nilai bobot sebelumnya
$b(t-1)$: nilai bobot sebelumnya
η : *learning rate*
$\xi(t)$: momentum

4. Ulangi tahap (2) dan (3) sampai mencapai sejumlah iterasi yang ditetapkan

F. *PEMBUATAN KODE BINER*[2]

Proses ini bertujuan untuk membuat kode biner untuk setiap video berdasarkan kode *hash keyframe*. Masukan untuk fungsi ini berupa kode *hash keyframe*. Variabel h_{il} menunjukkan kode biner *keyframe* ke-i dengan panjang kode sebesar l. Variabel Ind_i menunjukkan indeks *keyframe* pada video dan $|.|$ menunjukkan kardinalitas. Untuk membuat kode biner digunakan fungsi ambang $T(x)$ yang akan bernilai 1 jika $x > 0.5$ dan untuk nilai lainnya bernilai 0. Persamaan untuk membuat kode biner dapat dilihat pada persamaan (19).

$$h_{il} = T\left(\frac{1}{|\text{Ind}_i|}\sum_{j\in \text{Ind}_i} z_{jl}\right) \quad (19)$$

Matriks kode biner yang didapatkan akan berukuran k x 100 dengan nilai setiap elemennya antara 0 atau 1. Bagian baris berukuran k menunjukkan jumlah video, sedangkan bagian kolom berukuran 100 menunjukkan panjang kode biner 100 bit.

III. METODE PENELITIAN YANG DITAWARKAN

Pada penelitian ini diusulkan metode ekstraksi *keyframe* menggunakan gabungan dari *uniform time sampling* dan *shot boundary detection*. Metode ekstraksi *keyframe* yang diusulkan menggunakan algoritma jarak *hausdorff* dan deteksi tepi *canny*. Kemudian digunakan ciri LDP sebagai pengganti ciri LBP yang digunakan pada penelitian sebelumnya.

A. *JARAK HAUSDORFF*[9]

Jarak *Hausdorff* adalah metode untuk menghitung kemiripan antara dua bentuk poligon. Metode ini merupakan pengembangan dari metode jarak *euclidian*. Jarak *euclidian* dikembangkan karena metode ini memiliki beberapa kekurangan dalam menghitung kemiripan dua bentuk poligon. Kekurangannya adalah jarak *euclidian* terpendek tidak mempertimbangkan keseluruhan bentuk poligon, dan tidak memperhitungkan posisi rotasi dari poligon. Padahal dalam pencocokan dua bentuk citra, keseluruhan bentuk poligon dan posisi rotasi dari poligon merupakan informasi yang sangat menentukan kemiripan dua bentuk citra.

Hausdorff Distance dapat dihitung dengan persamaan berikut :

$$H(A,B) = \max\big(h(A,B), h(B,A)\big) \quad (20)$$

Dimana

$$h(A,B) = \underset{a\in A}{max}\ \underset{b\in B}{min}\ \|a-b\| \quad (21)$$

Keterangan :
A, B : hasil deteksi tepi *canny* terhadap suatu *frame* citra

B. *DETEKSI TEPI CANNY*[8]

Deteksi tepi pada citra merupakan proses untuk mencari perbedaan intensitas yang menyatakan batas-batas suatu objek dalam citra. Tujuan pendeteksian tepi adalah untuk meningkatkan penampakan garis batas suatu objek di dalam citra. Proses deteksi tepi citra dilakukan dengan mencari piksel-piksel yang memiliki nilai intensitas yang sangat berbeda dengan piksel-piksel yang berdekatan. Deteksi tepi *canny* adalah salah satu operator untuk proses deteksi tepi pada citra.

Berikut ini tahapan deteksi tepi *canny* :

1. Konvolusi citra dengan *Gaussian Filter*
   Merupakan proses penghalusan citra untuk menghilangkan noise.

2. Hitung *gradient magnitude* citra
Citra hasil penghalusan lalu dikonvolusi dengan Gx dan Gy. Gx adalah operator *sobel* horizontal dan Gy adalah operator *sobel* vertikal.

-1	0	+1
-2	0	+2
-1	0	+1

Gx

+1	+2	+1
0	0	0
-1	-2	-1

Gy

Gambar 1. *Sobel Mask*

Hasil dari kedua operator digabungkan dengan rumus:

$$G = \sqrt{{G_x}^2 + {G_y}^2} \quad (22)$$

Selanjutnya menentukan arah tepi dengan rumus :

$$\theta = \arctan\left(\frac{G_y}{G_x}\right) \quad (23)$$

Semua arah tepi yang berkisar antara 0 dan 22,5 serta 157,5 dan 180 derajat diubah menjadi 0 derajat. Semua arah tepi yang berkisar antara 22,5 dan 67,5 derajat diubah menjadi 45 derajat. Semua arah tepi yang berkisar antara 67,5 dan 112,5 derajat diubah menjadi 90 derajat. Semua arah tepi yang berkisar antara 112,5 dan 157,5 derajat diubah menjadi 135 derajat. Sehingga arah tepi hanya bernilai (0^o, 45^o, 90^o, 135^o)

3. *Non-maximum suppression*
Proses menghilangkan piksel-piksel yang tidak bernilai maksimum sehingga hanya nilai maksimum yang ditandai sebagai tepi. Hasilnya didapatkan garis tepi yang lebih ramping.

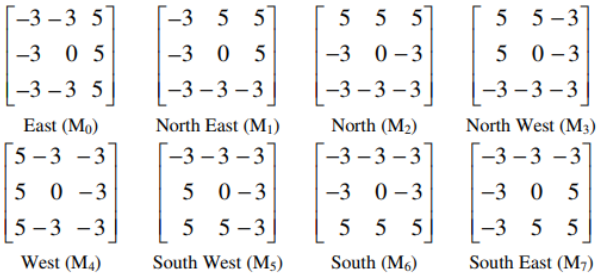
4. *Hystheresis Thresholding*
Jika nilai gradien > batas atas maka ditetapkan sebagai tepi kuat dan pikselnya dibuat bernilai '1' yang berarti piksel berwarna putih.

Jika nilai gradien < batas bawah maka ditetapkan bukan tepi dan pikselnya dibuat bernilai '0' yang berarti piksel berwarna hitam.

Jika nilai gradien diantara batas atas dan batas bawah maka akan dianggap sebagai tepi lemah. Bila tepi lemah ini terhubung dengan tepi kuat maka akan dianggap sebagai tepi, jika tidak maka akan dianggap sebagai bukan tepi.

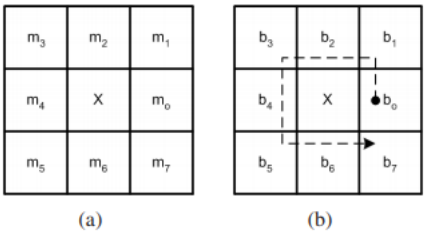
C. *LOCAL DIRECTIONAL PATTERN* (LDP)[6]

Local Directional Pattern adalah deskriptor lokal yang merepresentasikan tekstur pada citra. Pada deskriptor lokal, citra yang menjadi masukan akan dibagi menjadi beberapa bagian citra dengan ukuran yang lebih kecil dari ukuran citra masukan. Kemudian untuk setiap blok ukuran 3x3 piksel pada citra bagian akan dikonvolusikan dengan delapan arah kirsch *mask*. Delapan arah kirsch *mask* dapat dilihat pada Gambar 2.



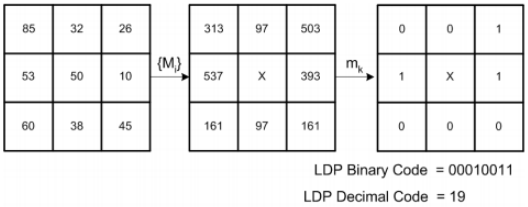
Gambar 2. Delapan Kirsch *Mask*

Hasil konvolusinya mendapatkan nilai respon tepi di kedelapan arah $\{m_i\}, i = 0,1,\ldots,7$ untuk piksel pada pusat blok 3x3 citra bagian. Posisi delapan nilai respon tepi kirsch *mask* pada blok citra ukuran 3x3 piksel dapat dilihat pada Gambar 3 (a), sedangkan untuk posisi bit biner LDP dapat dilihat pada Gambar 3 (b).



Gambar 3. (a) Posisi Delapan Tanggapan Tepi. (b) Posisi Bit Biner LDP [10]

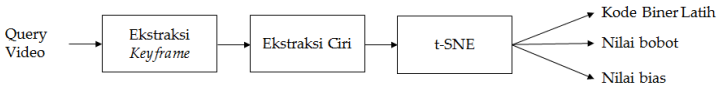
Nilai respon tepi dari kedelapan arah tidak semuanya penting, karena adanya sudut atau tepi akan memberikan nilai respon tepi yang tinggi dalam beberapa arah tertentu. Karena itu dipilih sejumlah ***k*** arah yang mempunyai nilai respon tepi tertinggi untuk menghasilkan kode LDP. Arah yang memiliki respon tepi tertinggi, maka bit binernya akan dibuat '1' sedangkan arah lainnya bit binernya akan dibuat '0'. Kemudian kode biner yang terbentuk dikonversi menjadi nilai desimal sehingga didapatkan kode LDP untuk piksel pada pusat blok 3x3 citra bagian. Proses ini pun dilakukan untuk semua citra bagian.



Gambar 4. Contoh Kode LDP untuk Nilai ***k*** = 3. [10]

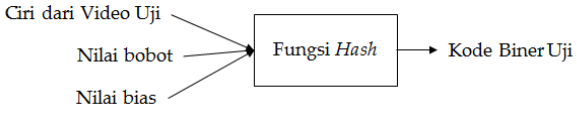
IV. PERANCANGAN SISTEM

Secara garis besar sistem NDVR dapat dibagi menjadi 3 diagram blok. Pada diagram blok pertama terdiri atas proses ekstraksi *keyframe* dari setiap video yang akan dilatih. Kemudian proses kedua adalah ekstraksi ciri dari setiap *keyframe* yang didapatkan menggunakan deskriptor lokal dan deskriptor global. Dan yang terakhir adalah proses pelatihan sistem menggunakan algoritma t-SNE. Proses ini dapat digambarkan dengan diagram blok pada Gambar 5.



Gambar 5. Diagram Blok Ekstraksi Ciri dan Pelatihan

Pada diagram blok kedua dilakukan proses untuk mendapatkan kode biner uji. Masukannya berupa ciri yang didapat dari 2 buah video uji serta nilai bobot dan bias yang didapatkan dari hasil diagram blok pertama.



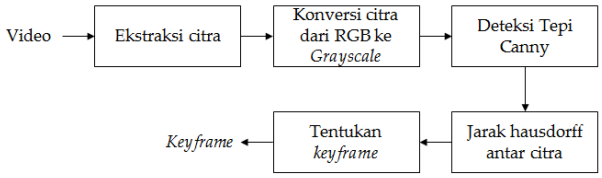
Gambar 6. Diagram Blok Pembuatan Kode Biner Uji

Pada diagram blok ketiga dilakukan proses pengujian antara kode biner latih dan kode biner uji menggunakan metode jarak *hamming*. Keluaran dari jarak *hamming* adalah jumlah perbedaan bit biner dan rank kemiripan video.

Gambar 7. Diagram Blok Pengujian

A. *EKSTRAKSI KEYFRAME*

Ekstraksi *keyframe* dilakukan untuk mendapatkan citra - citra yang dapat mewakili sebuah video. Untuk setiap detik pada sebuah video akan dipilih satu citra sebagai *keyframe* yang mewakili seluruh citra pada detik tersebut. *Keyframe* yang dipilih untuk setiap detik ini tetap memperhitungkan perubahan adegan antar detiknya. Proses ekstraksi *keyframe* dapat dilihat pada Gambar 8.



Gambar 8. Diagram Blok Ekstraksi *Keyframe*

Tahap pertama adalah masukan yang berupa video dilakukan ekstraksi citra, proses ini berguna untuk mengubah video menjadi kumpulan citra yang menyusunnya. Kemudian setiap citra yang didapatkan dikonversi dari model warna RGB menjadi *grayscale* yang memiliki nilai piksel antara 0 sampai 255 menggunakan fungsi MATLAB rgb2gray. Lalu, citra *grayscale* dilakukan proses deteksi tepi *canny*. Proses ini dilakukan untuk mendapatkan tepian dari seluruh objek yang terdapat pada citra.

Setelah didapatkan hasil deteksi tepi *canny* untuk seluruh citra pada video. Kemudian dipilih citra dengan indeks pertama pada detik ke-1 untuk dijadikan sebagai *keyframe* awal. Selanjutnya hitung jarak *hausdorff* antara hasil deteksi tepi *keyframe* awal dengan semua hasil deteksi tepi pada citra di detik ke-2. Setelah didapatkan jarak *hausdorff* untuk citra di detik ke-2 kemudian dilakukan pengurutan jarak *hausdorff* dari nilai terkecil ke terbesar.

Citra yang memiliki nilai jarak *hausdorff* terbesar lalu ditetapkan sebagai *keyframe* pada detik ke-2. Dipilihnya citra yang memiliki jarak *hausdorff* terbesar sebagai *keyframe* karena semakin besar nilai jarak *hausdorff*, maka menunjukkan kedua citra sangat berbeda dan secara tidak langsung menunjukkan adanya perubahan adegan yang cukup signifikan. Kemudian proses perhitungan jarak *hausdorff* ini dilakukan untuk mendapatkan *keyframe* pada detik ke-3 sampai detik terakhir video.

B. *EKSTRAKSI HSV*

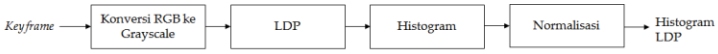
Proses untuk ekstraksi ciri HSV dapat dilihat pada diagram blok berikut ini :

Gambar 9. Diagram Blok Ekstraksi Ciri HSV

Kemudian citra HSV ini dilakukan kuantisasi warna menjadi histogram dengan ukuran 162 *bin*. Nilai *hue* dikuantisasi menjadi 18 *bin*, *saturation* menjadi 3 *bin*, sedangkan *value* dikuantisasi menjadi 3 *bin*. Sehingga akan didapatkan kombinasi sebanyak 18x3x3 = 162. Nilai *hue* dikuantisasi menjadi 18 *bin* karena *hue* memberikan informasi yang lebih banyak dan sistem visual manusia lebih sensitif terhadap perubahan *hue*. Setelah citra HSV dikuantisasi kemudian dibuat histogram dari nilai piksel HSV tersebut. Nilai dari histogram HSV ini pun dinormalisasi menjadi nilai antara 0 dan 1.

C. *EKSTRAKSI LDP*

Proses untuk ekstraksi ciri LDP dapat dilihat pada diagram blok berikut ini :



Gambar 10. Diagram Blok Ekstraksi Ciri LDP

Masukan untuk proses ekstraksi LDP adalah citra *keyframe* dengan model warna RGB yang telah dikonversi ke *grayscale*. Citra *grayscale* ini diubah ukurannya menjadi 100x100 piksel. Lalu, citra dibagi menjadi beberapa bagian dengan ukuran 25x25 piksel per bagiannya. Sehingga didapatkan 16 bagian citra dengan ukuran 25x25 piksel yang berasal dari sebuah citra ukuran 100x100 piksel. Kemudian untuk setiap blok ukuran 3x3 piksel pada citra bagian akan dikonvolusikan dengan delapan arah *kirsch mask*. Hasil yang didapatkan adalah nilai

respon tepi untuk delapan arah terhadap pusat piksel dari blok ukuran 3x3 piksel.

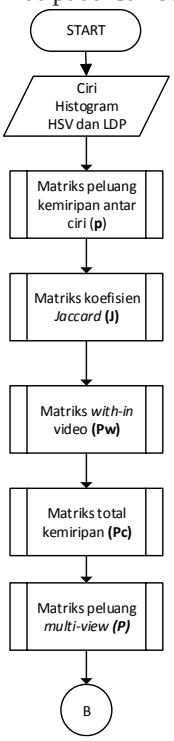
Pada penelitian ini, dipilih 3 arah yang mempunyai nilai respon tepi tertinggi. Proses pembuatan kode biner LDP adalah dengan membuat 3 arah yang memiliki nilai respon tertinggi menjadi bit biner '1', sedangkan untuk 5 arah lainnya dibuat menjadi bit biner '0'. Lalu kode biner yang didapat dikonversi menjadi kode desimal LDP. Sehingga didapatkan kode desimal LDP untuk pusat piksel dari sebuah blok ukuran 3x3, kemudian proses konvolusi diulang terhadap seluruh blok ukuran 3x3 piksel citra bagian. Lalu, kode LDP untuk seluruh piksel citra bagian dibuat menjadi histogram dengan ukuran 56 *bin*.

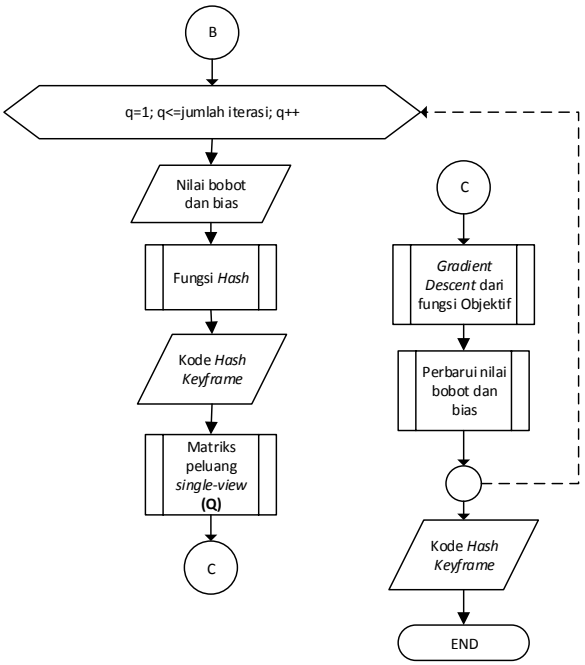
Proses perhitungan kode LDP serta histogram dilakukan untuk 16 bagian citra. Sehingga didapatkan 16 histogram yang merepresentasikan nilai kode LDP untuk setiap bagian citra yang memiliki ukuran 25x25 piksel. Kemudian 16 histogram ini digabung menjadi 1 histogram yang merepresentasikan sebuah citra *keyframe* yang berukuran 100x100 piksel. Cara menggabungkan 16 histogram ini adalah dengan menempelkan setiap histogram kesamping sehingga histogramnya menjadi ukuran 896 *bin*. Dan proses yang terakhir adalah histogram dinormalisasi sehingga nilainya berkisar antara 0 dan 1.

D. *t-Distributed Stochastic Neighbor Embedding* (t-SNE)[14]

Proses ini bertujuan untuk pembelajaran beberapa ciri serta untuk mereduksi dimensi data dengan memodelkan data berdimensi tinggi (*multi-view data*) ke berdimensi rendah (*single-view data*). Algoritma t-SNE ini dipilih karena kelebihannya dalam mengelompokkan data berdasarkan kelas dan dalam memvisualisasikan hasil dari pengelompokkan data tersebut.

Secara garis besar proses t-SNE yang digunakan dalam penelitian ini dapat dilihat pada Gambar 11.





Gambar 11. Diagram Alir t-SNE

E. *Pembuatan Kode Biner Latih dan Kode Biner Uji*

Pada pembuatan kode biner latih, kode *hash keyframe* yang didapatkan dari hasil pelatihan t-SNE kemudian dibuat kode binernya berdasarkan persamaan (19). Sedangkan pada pembuatan kode biner uji, masukannya berupa video uji untuk query video 3 dan query video 4. Video uji tersebut kemudian dilakukan proses ekstraksi *keyframe* lalu ekstraksi ciri. Sehingga didapatkan ciri berupa histogram HSV dan histogram LDP dari video uji.

Proses pembuatan kode biner uji terdiri dari 2 tahapan, yakni tahap pertama adalah perhitungan fungsi *hash* sesuai dengan persamaan (1). Dengan masukan berupa ciri histogram HSV, ciri histogram LDP, serta nilai bobot dan bias yang didapatkan setelah pelatihan t-SNE selesai. Keluaran dari fungsi *hash* adalah kode *hash keyframe* yang akan menjadi masukan pada tahap kedua. Pada tahap kedua dilakukan pembuatan kode biner dengan persamaan (19).

F. *Pencocokan Kode Biner Latih dan Kode Biner Uji*

Matriks kode biner latih yang didapatkan dari sistem NDVR yang dibuat akan berukuran 50x100. Baris ke-1 sampai 25 pada matriks merupakan kode biner latih untuk 25 video yang terdapat pada query video 3. Sedangkan baris ke-26 sampai 50 pada matriks merupakan kode biner latih untuk 25 video yang terdapat pada query video 4. Untuk kode biner uji query video 3 dan query video 4 akan berukuran 1x100 untuk setiap kode biner uji.

Proses pencocokan dilakukan terlebih dahulu pada query video 3. Proses pencocokan dilakukan dengan menghitung jarak *hamming* antara kode biner uji dan kode biner latih. Kode biner uji query video 3 dihitung jarak *hamming* terhadap matriks kode biner latih pada baris ke-1 sampai 25. Keluaran dari jarak *hamming* adalah jumlah perbedaan bit biner. Lalu,

dilakukan proses pengurutan rank video berdasarkan jumlah perbedaan bit biner dari nilai terkecil ke terbesar.

Proses pencocokan juga dilakukan pada query video 4 dengan menggunakan cara yang sama seperti pada query video 3. Hanya saja perhitungan jarak *hamming* dilakukan antara kode biner uji query video 4 terhadap matriks kode biner latih pada baris ke-26 sampai 50.

Keluaran terakhir dari sistem NDVR ini menghasilkan nilai jumlah perbedaan bit biner antara kode biner uji terhadap kode biner latih dan rank kemiripan video.

V. DATA PENGAMATAN

Pada penelitian ini pengambilan data dilakukan terhadap 25 video pada query video 3 dan 25 video pada query video 4. Pengambilan data dilakukan setelah proses pengujian antara video uji dengan 25 video pada query video 3 selesai diuji dan pengujian antara video uji dengan 25 video pada query video 4 selesai diuji. Pengambilan data dilakukan dengan tiga variasi ciri, tiga variasi jumlah iterasi, dan dua variasi jumlah video latih untuk setiap kelas. Variasi ciri terdiri dari ciri HSV, ciri LDP, dan gabungan ciri HSV dan LDP. Variasi jumlah iterasi terdiri dari 50 iterasi, 150 iterasi, dan 200 iterasi.

Variasi jumlah video latih untuk setiap kelas terdiri dari tipe *imbalance* query video dan tipe *balance* query video. Untuk tipe *imbalance* query video terdiri dari 5 *near duplicate video* (NDV) dan 20 *dissimilar video* (DV), sedangkan untuk tipe *balance* query video terdiri dari 12 *near duplicate video* (NDV) dan 13 *dissimilar video* (DV). Sehingga untuk masing – masing tipe *imbalance* dan *balance* terdapat 9 kombinasi pengambilan data sebagai berikut :

Tabel 1. Kombinasi Pengambilan Data

Tipe *imbalance* query video	Tipe *balance* query video
Ciri HSV dengan 50 iterasi	Ciri HSV dengan 50 iterasi
Ciri HSV dengan 150 iterasi	Ciri HSV dengan 150 iterasi
Ciri HSV dengan 200 iterasi	Ciri HSV dengan 200 iterasi
Ciri LDP dengan 50 iterasi	Ciri LDP dengan 50 iterasi
Ciri LDP dengan 150 iterasi	Ciri LDP dengan 150 iterasi
Ciri LDP dengan 200 iterasi	Ciri LDP dengan 200 iterasi
Ciri (HSV+LDP) dengan 50 iterasi	Ciri (HSV+LDP) dengan 50 iterasi
Ciri (HSV+LDP) dengan 150 iterasi	Ciri (HSV+LDP) dengan 150 iterasi
Ciri (HSV+LDP) dengan 200 iterasi	Ciri (HSV+LDP) dengan 200 iterasi

A. *HASIL PENGAMBILAN DATA*

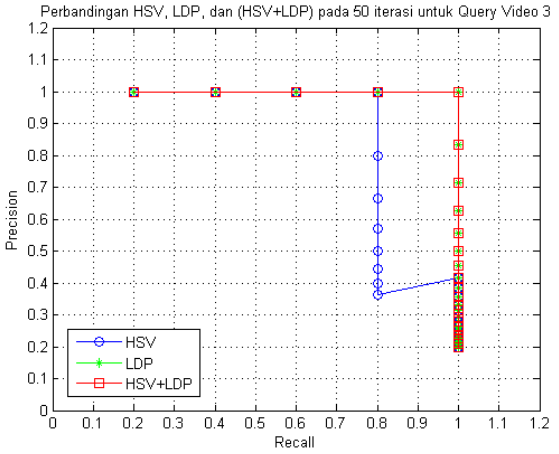
Data pengamatan yang diambil berupa tabel hasil deteksi duplikasi video dengan variasi yang sesuai dengan kombinasi pengambilan data pada Tabel 1. Kolom ke-1 pada tabel artinya urutan rank kemiripan query video terhadap video uji dari query tersebut. Kolom ke-2 pada tabel menunjukkan jumlah perbedaan bit *Hamming*. Kolom ke-3 menunjukkan indeks dari video latih. Kolom ke-4 terdapat istilah Prec yang artinya *Precision*. Kolom ke-5 terdapat istilah Rec yang artinya *Recall*.

Tabel 2. Hasil Deteksi Duplikasi Video untuk *Imbalance* Query Video pada Ciri HSV dengan 50 Iterasi

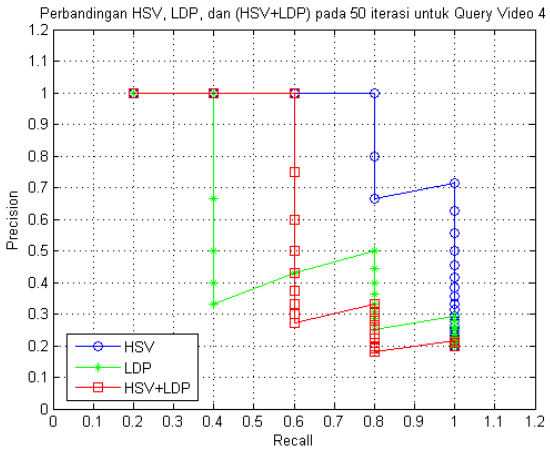
Rank	Ciri HSV dengan 50 iterasi							
	Query video 3				Query video 4			
	Jarak Hamming (bit)	Indeks video	Prec	Rec	Jarak Hamming (bit)	Indeks Video	Prec	Rec
1	22	5	1	0.200	11	4	1	0.200
2	25	3	1	0.400	16	5	1	0.400
3	26	1	1	0.600	19	2	1	0.600
4	26	2	1	0.800	28	3	1	0.800
5	39	8	0.800	0.800	28	16	0.800	0.800
6	39	11	0.667	0.800	34	25	0.667	0.800
7	40	10	0.571	0.800	35	1	0.714	1
8	40	21	0.500	0.800	37	24	0.625	1
9	42	23	0.444	0.800	38	19	0.556	1
10	43	25	0.400	0.800	39	17	0.500	1
11	46	12	0.364	0.800	40	8	0.455	1
12	48	4	0.417	1	41	15	0.417	1
13	49	7	0.385	1	42	6	0.385	1
14	49	20	0.357	1	43	18	0.357	1
15	50	6	0.333	1	44	21	0.333	1
16	50	9	0.313	1	44	22	0.313	1
17	50	13	0.294	1	46	10	0.294	1
18	50	14	0.278	1	46	23	0.278	1
19	50	15	0.263	1	47	7	0.263	1
20	50	17	0.250	1	47	9	0.250	1
21	50	18	0.238	1	47	13	0.238	1
22	50	19	0.227	1	49	14	0.227	1
23	50	22	0.217	1	52	11	0.217	1
24	50	24	0.208	1	52	12	0.208	1
25	52	16	0.200	1	52	20	0.200	1

Berdasarkan tabel 2, pada query video 3 dan 4 terlihat bahwa pada rank ke-1 sampai 4 memiliki nilai *precision* sebesar 1. Hal ini menunjukkan sistem berhasil mendeteksi NDV berturut-turut pada rank tersebut. Nilai *recall* pada query video 3 baru bernilai 1 pada rank ke-12 ini menunjukkan sistem baru berhasil mendeteksi semua NDV pada rank ke-12. Sedangkan nilai *recall* pada query video 4 baru bernilai 1 pada rank ke-7 ini menunjukkan sistem baru berhasil mendeteksi semua NDV pada rank ke-7.

Berdasarkan nilai *precision* dan *recall* yang didapat dari tabel deteksi duplikasi video kemudian dibuat grafik *precision-recall* untuk semua kombinasi pengambilan data seperti pada contoh Gambar 12 dan Gambar 13.



Gambar 12. Grafik *Precision-Recall* untuk *Imbalance* Query Video 3 pada 50 Iterasi



Gambar 13. Grafik *Precision-Recall* untuk *Imbalance* Query Video 4 pada 50 Iterasi

B. *ANALISA DATA*

Pada aplikasi *Near Duplicate Video Retrieval* (NDVR) ini, kinerja sistem diukur dari 2 parameter yaitu *precision* dan *recall*. Pada penelitian ini diinginkan nilai *precision* dan *recall* yang sebaik mungkin, hal ini digambarkan dengan grafik *precision-recall* yang semakin mendekati titik (1,1). Kemudian dihitung nilai *Average Precision* (AP) pada *imbalance* query video dan *balance* query video serta nilai *Mean Average Precision* (MAP). Berikut ini akan ditampilkan rangkuman dari hasil percobaan dan hasil rangkumannya dapat dilihat pada tabel 3, tabel 4, tabel 5, dan tabel 6.

Tabel 3. Rangkuman Nilai *Average Precision* (AP) untuk setiap *Imbalance* Query Video

Iterasi	HSV		LDP		HSV+LDP	
	Query video 3	Query video 4	Query video 3	Query video 4	Query video 3	Query video 4
50	0.883	0.943	1	0.645	1	0.710
150	0.856	0.925	1	0.648	0.967	0.756
200	0.967	0.848	1	0.648	1	0.777

Tabel 4. Rangkuman Nilai *Average Precision* (AP) untuk setiap *Balance* Query Video

Iterasi	HSV		LDP		HSV+LDP	
	Query video 3	Query video 4	Query video 3	Query video 4	Query video 3	Query video 4
50	0.895	0.962	0.987	0.875	0.971	0.927
150	0.895	0.967	1	0.779	1	0.918
200	0.890	0.960	0.987	0.783	1	0.913

Tabel 5. Rangkuman Nilai *Mean Average Precision* (MAP) untuk setiap *Imbalance* Query Video

Iterasi	HSV	LDP	HSV+LDP
50	0.913	0.822	0.855
150	0.890	0.824	0.861
200	0.907	0.824	0.888

Tabel 6. Rangkuman Nilai *Mean Average Precision* (MAP) untuk setiap *Balance* Query Video

Iterasi	HSV	LDP	HSV+LDP
50	0.928	0.931	0.949
150	0.931	0.890	0.959
200	0.925	0.885	0.956

Jika dilihat dari data tabel 5 pada ciri HSV didapatkan nilai MAP yang tertinggi untuk semua variasi iterasi. Selain itu jika dilihat pada ciri LDP didapatkan nilai MAP yang terendah untuk semua variasi iterasi. Sedangkan pada penggunaan gabungan ciri HSV dan LDP didapatkan nilai MAP yang lebih baik dibandingkan penggunaan ciri LDP saja namun nilai MAP nya masih belum lebih baik daripada penggunaan ciri HSV saja.

Kemudian jika dilihat dari data tabel 6 pada ciri HSV terdapat sedikit kenaikkan nilai MAP dibandingkan pada tabel 5 dengan kenaikkan 0.015 untuk 50 iterasi, 0.041 untuk 150 iterasi dan 0.018 untuk 200 iterasi. Namun pada ciri LDP terdapat kenaikkan nilai MAP yang cukup besar yakni 0.109 untuk 50 iterasi, 0.066 untuk 150 iterasi dan 0.061 untuk 200 iterasi.

Dan pada penggabungan ciri HSV dan LDP terjadi kenaikkan nilai MAP yang sangat besar yakni 0.094 untuk 50 iterasi, 0.098 untuk 150 iterasi dan 0.068 untuk 200 iterasi. Selain itu penggunaan gabungan ciri HSV dan LDP pada kasus *balance* query video menghasilkan nilai MAP yang lebih baik dibandingkan penggunaan ciri HSV saja atau ciri LDP saja.

Dari data nilai MAP pada tabel 6 terlihat bahwa nilai MAP tertinggi didapatkan untuk 150 iterasi, sedangkan pada 200 iterasi nilai MAP mengalami penurunan nilai yang sangat kecil berkisar 0.005. Hal ini dikarenakan pada awal pelatihan dilakukan inisialisasi nilai bobot dan bias secara acak yang mengakibatkan setiap kali pelatihan nilai MAP yang didapatkan dapat berubah-ubah namun perubahannya sangat kecil.

Berdasarkan data nilai MAP di tabel 5 dan tabel 6 terlihat bahwa *balance* query video mempunyai nilai MAP yang lebih baik dibandingkan dengan *imbalance* query video. Hal ini berarti bahwa jumlah video latih untuk setiap kelas sangat mempengaruhi kinerja sistem *Near Duplicate Video Retrieval* (NDVR).

VI. KESIMPULAN

Nilai MAP untuk *balance* query video lebih besar daripada *imbalance* query video pada semua variasi iterasi dan semua variasi ciri. Untuk sistem dengan jumlah video latih yang sama untuk tiap kelas akan menghasilkan nilai MAP yang lebih baik jika menggunakan gabungan ekstraksi ciri lokal dan

global dibandingkan dengan hanya satu ekstraksi ciri (lokal saja atau global saja). Dalam pelatihan sistem sebaiknya berdasarkan nilai fungsi objektif yang diinginkan daripada menggunakan jumlah iterasi agar pelatihan sistem lebih optimal. Sedangkan untuk mengatasi *imbalance* query video perlu ditambahkan metode *cost-sensitive learning* untuk mengurangi kesalahan dalam klasifikasi.

REFERENSI

[1] Hao, Y., T. Mu., R. Hong., M. Wang., N. An. and J.Y. Goulermas.2017. Stochastic Multiview Hashing for Large-Scale Near-Duplicate Video Retrieval. *IEEE Trans. Multimedia*.Vol19No 1 : 1–14.

[2] Hao, Y., T. Mu., R. Hong, M. Wang, N. An. and J. Y. Goulermas.2017. Unsupervised t-Distributed Video Hashing and Its Deep Hashing Extension. *IEEE Trans. Image Processing*. Vol26No 11 : 5531–5544.

[3] Patel, B.V. and B.B. Meshram.2012. Content Based Video Retrieval. *The International Journal of Multimedia & Its Application (IJMA)*. Vol 4 No 5 : 77-98.

[4] Khurana, M.K. and Dr.M.B. Chandak. 2013. Key Frame Extraction Methodology for Video Annotation. *IJCET*. Vol 4 No. 2 : 221-228.

[5] Vimalraj, S.S. and R. Porkodi. 2018. A Review on Handling Imbalanced Data. *2018 International Conference on CurrentTrends towards Converging Technologies* (ICCTCT). Pp 1-11.

[6] Jabid, T., Md.H. Kabir. and O. Chae. 2010. Local DirectionalPattern (LDP) for Face Recognition. *2010 Digest of Technical Papers International Conference on Consumer Electronics (ICCE)*.pp329-330.

[7] Wu, X., A.G. Hauptmann. and C.W. Ngo. 2007. PracticalElimination of Near-Duplicates from Web Video Search. *Proceedings of the15th ACM International Conference on Multimedia*. pp 218-227.

[8] Al-amri, S.S, N.V. Kalyankar. and S.D. Khamitkar. 2010. Image Segmentation By Using Edge Detection. *International Journal on Computer Science and Engineering* (IJCSE). Vol 2 No 3 : 804-807.

[9] Huttenlocher, D.P., G.A. Klanderman. and W.J. Rucklidge. 1993. Comparing Images Using the Hausdorff Distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Vol 15No9:850-863.

[10] Kabir, Md.H., T. Jabid. and O. Chae. 2010. A Local Directional Pattern Variance (LDPv) Based Face Descriptor for Human Facial Expression Recognition. *2010 7th IEEE InternationalConferenceon Advanced Video and Signal Based Surveillance*. pp 526-532.

[11] Plataniotis, K.N. and A.N. Venetsanopoulos. 2000. Color Image Processing and Applications. Springer : Verlag.

[12] Gonzalez, R.C., R.E. Woods. 2008. Digital Image Processing. 3rd Edition. New Jersey : Prentice Hall.

[13] Stallings, W. 2014. Cryptography and Network Security Principles and Practice. 6th Edition. Pearson Education,Inc.

[14] Maaten, L. and G. Hinton. 2008. Visualizing High-DimensionalData Using t-SNE. *Journal of Machine Learning Research*. pp 2579-2605.

[15] Hamming, R.W., 1950. *Error Detecting and Error Correcting Codes*. *Bell Labs Technical Journal*, Vol 29 No 2 : 147-160.

[16] Powers, D.M.W. 2007. Evaluation : From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation.Technical Report SIE-07-001. School of Informatics and Engineering,Flinders University of South Australia.

[17] Manning, C.D., P. Raghavan., H.Schutze. 2008. Introduction to Information Retrieval. Cambridge University Press.