APLIKASI MOBILE BERBASIS CBIR UNTUK PENCARIAN PRODUK PONSEL PADA ONLINESHOP

Nickolas Cornelius Siantar [1] Janson Hendryli [2] Dyah Erny Herwindiati [3]
[1), 2), 3)] Teknik Informatika, FTI, Universitas Tarumanagara
Jl. Letjen S Parman no 1, Jakarta 11440 Indonesia
[1)] nikolas.535150077@stu.untar.ac.id, [2)] jansonh@fti.untar.ac.id, [3)] dyahh@fti.untar.ac.id

ABSTRACT

Phone or smartphone and online shop, there is something that cannot be separated with human. There are so many type of smartphones show up in the market that people are confused on which one to get on the online stores. Smartphones recognition is done by using the Histogram of Oriented Gradient to recognize shapes of phones, Color Quantization to recognize the color, and Local Binary Pattern to recognize texture of the phones. The output of the Feature Extractor is a feature vector which is used on the LVQ to process recognize through finding the smallest Euclidean Distance between the trained vectors. The result of this paper is an application that can recognize 16 phone types using the image with the accuracy of 9.6%

Key words

Color Quantization, Histogram of Oriented Gradient, Learning Vector Quantization, Local Binary Pattern, Smartphone.

1. Pendahuluan

Seiring dengan perkembangan teknologi informasi (TI) yang sangat pesat, kebutuhan manusia pun terus meningkat. Manusia selalu mencari sesuatu untuk dapat menjalankan kehidupannya dengan lebih mudah. Banyak penemuan pada bidang TI yang memberi berbagai kemudahan bagi manusia. Dengan demikian, manusia dapat menggunakan waktunya secara maksimal dan efisien.

Salah satu penemuan teknologi informasi yang memberikan kemudahan bagi manusia adalah mesin pencari. Teknologi ini sangat mempermudah manusia untuk mencari informasi yang dibutuhkannya dalam memenuhi kebutuhan sehari-hari. Mesin pencari berpengaruh besar dalam berbagai bidang kehidupan manusia, contohnya dalam bidang edukasi, ekonomi, sosial budaya, dan berbagai bidang lainnya. Namun, manusia selalu menginginkan hal lebih dari yang sudah ada, sehingga mesin pencari ini terus dikembangkan.

Mesin pencari dikembangkan untuk dapat digunakan dengan semudah mungkin, salah satunya yang sedang terus dikembangkan adalah mesin pencari menggunakan gambar atau foto atau dikenal juga dengan mesin pencari visual. Mesin pencari visual pertama kali diluncurkan oleh TinEye pada tahun 2008 dan pada tahun 2010 Google meluncurkan Google Goggles yang menjadikan pencarian dapat dilakukan langsung menggunakan kamera telepon seluler (ponsel). Mesin pencarian visual yang dimiliki TinEye pada saat itu hanya mampu melakukan pencarian mengenai landmark suatu negara namun tidak dapat mendeteksi garis tepi dari objek [1].

Content-Based Image Retrieval (CBIR) adalah sistem untuk melakukan pencarian dengan melakukan deteksi pada objek dalam citra digital yang merupakan suatu kemajuan dalam perangkat lunak mesin pencari. Awalnya CBIR ditujukan untuk mengatasi permasalahan munculnya koleksi gambar dalam skala besar. Pada CBIR, citra secara otomatis akan diindeks dengan melakukan peringkasan konten visual berdasarkan fitur yang diekstrak seperti warna, bentuk, dan tekstur [2].

Pada penulisan makalah ini akan dibahas mengenai aplikasi mobile berbasis CBIR untuk melakukan pencarian produk ponsel pada aplikasi toko online dengan menggunakan metode Histogram of Oriented Gradient (HOG), Color Quantization, dan Local Binary Pattern (LBP) sebagai feature extraction dan metode Learning Vector Quantization (LVQ) sebagai metode pembelajaran dan pengklasifikasian. Masukan perangkat lunak ini berupa citra digital RGB dan keluarannya berupa hasil pencarian produk yang terdapat pada toko online.

Metode HOG digunakan untuk melakukan pengambilan fitur bentuk dari citra. Langkah yang dilakukan pada metode HOG yaitu preprocessing, pada tahap ini citra disiapkan untuk diolah seperti pengaturan ukuran citra dan selanjutnya citra akan dihitung nilai besaran dan nilai arahnya. Citra yang sudah memiliki kedua nilai tersebut dibagi menjadi beberapa sel 8x8 untuk dilakukan proses pembentukan histogram, setelah histogram terbentuk maka dilakukan normalisasi lalu seluruh histogram yang membentuk vektor digabungkan membentuk vektor fitur raksasa. Pada metode Color Quantization, warna tertentu akan dijadikan acuan untuk pengelompokan warna. Nilai warna pada citra akan dihitung menggunakan euclidean distance untuk mencari jarak terdekat antara nilai warna dengan nilai warna acuan. Nilai warna yang memiliki jarak terdekat dengan nilai acuan, akan diambil dan dimasukan ke dalam kelompok nilai acuan tersebut. Hasil dari proses tersebut

akan membentuk histogram sehingga dapat diketahui komposisi warna pada citra tersebut. Model warna yang digunakan pada umumnya adalah model warna RGB. Pada metode LBP, fitur yang akan diambil merupakan tekstur dari citra. Pertama citra dikonversikan menjai citra grayscale, lalu citra akan diproses dengan operator LBP. Pada operator LBP, 8 nilai piksel pada sel berukuran 3x3 akan dibandingan dengan nilai piksel pusat lalu akan dilakukan proses konversi kode biner. Kode biner ini memiliki nilai desimal yang akan disimpan pada matriks baru berukuran sesuai citra semula dan histogram yang akan membentuk vektor fitur. Hasil dari ketiga metode tersebut akan digabungkan menjadi suatu vektor fitur yang akan digunakan menjadi vektor masukan pada metode LVQ.

Pada metode LVQ, akan dilakukan pelatihan terlebih dahulu secara terawasi dengan menggunakan vektor fitur yang ada. Setelah melakukan pelatihan, jaringan LVQ mampu untuk menentukan suatu objek. Pelatihan berupa penyesuaian bobot untuk setiap koneksi dan juga melatih jaringan agar dapat mengenali suatu pola pada citra yang telah dikumpulkan dalam vektor fitur. Bobot yang telah dilatih akan disimpan pada file csv untuk digunakan pada saat pengujian. Hasil dari pengenalan berupa ID kelas yang akan dibaca sistem dan mengambil data pada basis data berupa alamat website dari produk yang dijual, harga produk pada setiap toko online, dan nama toko online. Pengujian dilakukan dengan memberi masukan berupa citra dari kamera ponsel yang akan dikirimkan ke sistem REST. Citra yang diterima sistem REST akan diproses melalui *feature extraction* sehingga menghasilkan suatu vektor fitur dan akan dihitung jaraknya dengan bobot yang sudah dilatih. Bobot yang memiliki jarak minimum dengan vektor masukan akan dipilih. Hasil pengenalan akan dikirim untuk memanggil data pada basis data dan dikirimkan oleh sistem REST kembali ke android untuk ditampilkan pada modul keluaran.

2. Dasar Teori

2.1 Citra Digital

Citra digital merupakan representasi dari gambar nyata sebagai serangkai angka yang dapat disimpan dan ditangani oleh komputer digital. Untuk menerjemahkan gambar menjadi angka, citra dibagi menjadi area kecil yang disebul piksel (elemen gambar). Untuk setiap piksel, alat perekam citra mencatat angka atau sekumpulan angka kecil yang mendeskripsikan beberapa properti piksel ini, seperti kecerahanan (intensitas cahaya) atau warnanya. Angka-angka disusun dalam array baris dan kolom yang sesuai dengan posisi vertikal dan horizontal dari piksel dalam gambar [3]. Kualitas suatu citra dipengaruhi oleh jumlah piksel dan variasi niai piksel yang disebut dengan resolusi. Resolusi dinyatakan dalam jumlah piksel per inci (ppi). Semakin tinggi resolusi suatu citra maka semakin tinggi pula ketajaman atau detail citra tersebut.

2.2 Basis Data

Basis data merupakan kumpulan informasi saling terkait yang disusun sehingga dapat dengan mudah diakses, diatur dan diperbarui. Data disusun dalam baris, kolom, tabel, dan diindeks untuk mempermudah ketika melakukan pencarian informasi tersebut. Data dapat diperbarui, diperluas, dan dihapus saat informasi baru ditambahkan [4].

Basis data digunakan untuk menyimpan data yang akan digunakan dan diolah kembali. Terdapat beberapa hal yang perlu diketahui pada basis data. Setiap basis data memiliki tabel berisikan atribut yang merupakan nama dari kolom. Tuple adalah pertemuan antara baris dan kolom. Kardinalitas merupakan jumlah tuple pada suatu tabel relasi. Derajat adalah jumlah atribut dalam suatu tabel. Pada basis data, salah satu dari atribut harus berupa primary key agar data masukkan memiliki identitas sendiri-sendiri yang membedakan dengan data yang lain.

2.3 Sistem REST

REST atau REpresentational State Transfer adalah arsitektur untuk menyediakan suatu standar antara sistem komputer pada web, sehingga memudahkan sistem untuk saling berkomunikasi satu sama lain. Pada arsitektur REST, klien mengirimkan permintaan untuk mencari atau mengubah sumber daya dan server mengirimkan respon terhadap permintaan tersebut [5]. REST menggunakan JSON untuk mewakili sumber daya seperti teks. Permintaan biasanya terdiri dari :

1. Kata kerja HTTP yang mendefinisikan operasi yang perlu dilakukan.
2. Sebuah header, yang memungkinkan klien untuk menyampaikan informasi tentang permintaan tersebut.
3. Alamat menuju resource
4. Badan pesan yang berisi data bersifat opsional.

Terdapat 4 kata kerja HTTP yang digunakan pada proses permintaan untuk dapat berinteraksi dengan menggunakan sistem REST :

1. GET , untuk mengambil sumber daya tertentu berdasar ID.
2. POST, untuk membuat sumber daya baru.
3. PUT, untuk memperbarui sumber daya tertentu berdasar ID.
4. DELETE, untuk menghapus sumber daya tertentu berdasar ID.

2.4 JSON

JSON atau JavaScript Object Notation merupakan format pertukaran data berbasis teks yang dapat dibaca manusia yang digunakan untuk merepresentasikan struktur dan objek sederhana [6]. JSON digunakan sebagai alternatif XML dalam mengatur data. JSON adalah Bahasa yang independen dan dapat dikombinasikan dengan C++, Python, Java, Perl dan

lainnya. Dokumen JSON relative ringan dan dapat dieksekusi dengan cepat oleh web server.

Secara struktur, JSON terdiri dari dua struktur yaitu, kumpulan pasangan nama atau nilai dan daftar nilai yang terurut. Dalam berbagai Bahasa pemrograman, kumpulan pasangan nama atau nilai dapat berupa objek, struct, dictionary, tabel hash, keyed list, atau array asosiatif. Sementara daftar nilai dapat berupa vektor, array, list, atau sequence.

2.5 *Content-Based Image Retrieval*

Content-Based Image Retrieval (CBIR) adalah sistem pencarian citra berdasarkan konten atau fitur visual seperti warna, bentuk, dan tekstur atau struktur gambar [7].

Pada metode lama, pengindeksan citra dimulai dari menyimpan citra pada database dan mengaitkannya dengan kata kunci, mengaitkannya dengan deskripsi yang dikategorikan ini memakan waktu. Dalam CBIR, setiap citra yang disimpan dalam basis data memiliki fitur yang dibandingkan dengan fitur citra lainnya. Hal ini melibatkan dua langkah:

1. Feature Extraction, yaitu proses mengekstrasi fitur pada citra ketingkat yang dapat dibedakan.
2. Matching, merupakan proses pencocokan fitur-fitur untuk menghasilkan hasil yang serupa secara visual.

2.6 *Histogram of Oriented Gradient*

Histogram of Oriented Gradient (HOG) merupakan suatu metode yang digunakan untuk mengambil fitur berupa distribusi arah gradien dari suatu gambar. Dalam HOG gradien dari suatu gambar sangat berguna sebagai fitur untuk mengenali objek karena nilai gradien yang sangat besar terdapat di sekitar sudut atau tepi objek. Sehingga banyak informasi yang dapat diambil untuk diproses untuk mengetahui bentuk objek daripada gradien yang cenderung sama pada daerah datar.

Berikut proses yang dilakukan dalam HOG [8]:

1. Tahapan pertama ada preprocessing, yaitu melakukan resize pada citra menjadi ukuran 64 x 128. Pada tahap ini terdapat kendala berupa aspek rasio pada citra yang beragam. Untuk menghasilkan hasil deteksi yang baik, maka dilakukan cropping agar citra memiliki aspek rasio 1:2, misalnya 100:200, 128:256, atau 1000:2000.
2. Setelah citra berukuran 64 x 128, maka akan dilakukan proses perhitungan gradien vektor pada setiap piksel. Untuk menghitung gradien vektor, perlu terlebih dahulu menghitung gradien horizontal dan gradien vertikal. Perhitungan dapat dilakukan dengan mudah menggunakan filter dengan Sobel operator atau kernel berikut.

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

Gx

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

Gy

Gambar 1. Sobel operator

Setelah mendapatkan nilai gradien horizontal dan vertikal, maka selanjutnya dapat menghitung nilai gradien vektor dan arahnya menggunakan rumus berikut.

$$g = \sqrt{g_x^2 + g_y^2} \quad \text{.....(1)}$$

$$\theta = arctan\frac{g_y}{g_x} \quad \text{.....(2)}$$

Keterangan:
g = Nilai gradien
x, y = koordinat horizontal dan vertikal citra
θ = arah dari gradien

Hasil dari perhitungan tersebut menghasilkan citra gradien yang menghapus banyak informasi non-esensial seperti latar belakang yang berwarna konstan.

3. Selanjutnya menghitung Histogram of Oriented Gradients dengan memasukan citra ke sel 8x8. Citra yang dimasukan ke sel 8x8 akan memberikan representasi yang ringkas. Representasi ini selain lebih ringkas, representasi ini juga lebih kuat terhadap gangguan atau noise. Setiap gradien sel mengandung 2 nilai yaitu besaran dan arah per piksel sehingga terdapat 8x8x2 = 128 angka. Pada akhir proses ini, 128 angka tersebut akan disimpan pada histogram 9 wadah. Histogram yang digunakan merupakan sebuah vektor atau array 9 wadah yang berisikan sudut 0, 20, 40, 80, 100, 120, 140, dan 160. Setiap sel 8x8 mengandung 2 nilai tersebut akan dimasukan ke histogram 9 wadah dengan menyesuaikan nilai gradien arah.
4. Tahap berikutnya yaitu normalisasi blok 16x16, hal ini ditujukan agar histogram tidak terpengaruh dengan variasi pencahayaan karena pada tahap sebelumnya gradien citra cenderung sensitif terhadap efek pencahayaan secara keseluruhan. Jika citra dibuat lebih gelap dengan cara membagi 2 setiap nilai pada citra, maka nilai besaran gradien akan terbagi 2 juga dan nilai histogram akan terbagi pula. Maka normalisasi vektor ini dilakukan untuk menghilangkan skala. Blok 16x16 memiliki 4 histogram yang dapat digabungkan untuk membentuk vektor elemen 36x1 dan dapat dinormalisasikan seperti normalisasi vektor 3x1.

Lalu blok 16x16 dipindahkan 8 piksel berikutnya dan proses akan dilakukan berulang.

5. Proses terakhir pada HOG adalah menghitung nilai vektor fitur HOG. Untuk menghitung vektor fitur akhir seluruh bagian citra, seluruh vektor 36x1 digabungkan menjadi vektor raksasa. Besar dari vektor tersebut dapat dihitung sebagai berikut. Banyak posisi dari blok 16x16 yang terdapat pada citra yaitu 7 pada horizontal dan 15 pada vertikal sehingga totalnya ada 105 posisi. Setiap blok 16x16 diwakili oleh 36x1 vektor, maka ketika digabungkan akan didapatkan pangjang vektor 3780.

2.7 *Color Quantization*

Metode Color Quantization digunakan untuk mengambil fitur warna dan memperkecil jumlah dari kemungkinan kombinasi warna. Kombinasi warna pada RGB dengan rentang nilai dari 0 sampai 255 adalah 16.777.216. Hal tersebut akan memakan waktu untuk diproses. Color Quantization untuk proses ini menggunakan K-Means untuk melakukan pengelompokan warna. Berikut tahapan untuk proses Color Quantization dengan menggunakan K-Means [9]:

1. Melakukan identifikasi jumlah klaster yang diperlukan. Jumlah klaster akan menentukan nilai dari K.
2. Tentukan nilai dari K yang merepresentasikan sebuah warna yang akan digunakan.
3. Menghitung jarak seluruh nilai warna dengan nilai K. Proses perhitungan jarak dapat menggunakan euclidean distance.

$$D = \sqrt{(R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2} \quad \ldots\ldots(3)$$

Keterangan:
D = nilai *euclidean distance*
$R_{1,2}$ = nilai warna merah pada pixel citra dan K
$G_{1,2}$ = nilai warna hijau pada pixel citra dan K
$B_{1,2}$ = nilai warna biru pada pixel citra dan K

4. Mengklasifikasikan setiap nilai warna dengan jarak terdekat terhadap nilai K.

Dengan menggunakan color quantization, fitur warna dapat dikelompokan secara efektif tanpa mengganggu kinerja sistem secara keseluruhan.

2.8 *Local Binary Pattern*

Local Binary Pattern (LBP) digunakan untuk mengambil fitur berupa tekstur objek. Sebelum melakukan proses LBP, warna citra digital dikonversikan menjadi citra grayscale dan dilakukan penambahan padding. Selanjutnya proses yang dilakukan dalam LBP yaitu membandingkan nilai piksel dengan 8 piksel tetangganya pada sel berukuran 3x3. Nilai piksel kedelapan tetangganya akan dikurangi dengan nilai piksel pusat lalu akan diubah menjadi nilai 0 jika hasilnya negatif dan 1 jika hasil pengurangan positif. Selanjutnya akan diperoleh nilai biner dengan menggabungkan seluruh kode biner searah jarum jam mulai dari yang paling kiri atas dan ini berkaitan dengan nilai desimal yang nantinya akan dihasilkan. Kode biner yang diperoleh akan diubah menjadi nilai desimal. Hasil dari nilai desimal akan disimpan dalam histogram grayscale dan disimpan pada matriks baru dengan ukuran yang sesuai dengan citra semula.

2.9 *Learning Vector Quantization*

Learning Vector Quantization (LVQ) adalah jaringan saraf tiruan yang menggunakan pembelajaran supervised atau pembelajaran yang diawasi. LVQ menggunakan pembelajaran yang diawasi sehingga jaringannya akan diberikan pola pelatihan dengan pengklasifikasian yang sudah dikenali bersama dengan hasil dari outputnya. Setelah melakukan proses pelatihan, LVQ akan mengklasifikasikan vektor masukan dengan menugaskannya ke kelas yang sama dengan unit keluaran. Klasifikasi dilakukan dengan cara menemukan unit keluaran yang paling dekat dengan vektor masukan.

Secara arsitektur, LVQ mempunyai kemiripan dengan arsitektur Kohonen Self-Organizing Maps (KSOM). Namun, Arsitektur dapat dilihat pada gambar 6. Parameter yang digunakan pada proses pelatihan yaitu:

X = vektor pelatihan (x1, x2, …, xn)
T = kelas untuk pelatihan vektor x
wJ = bobot vektor untuk keluaran unit ke J
CJ = kelas untuk hasil keluaran ke J
// x-w_j // = Euclidean distance antara vektor masukan dengan bobot vektor untuk unit keluaran ke j

Berikut adalah algoritma yang dilakukan pada LVQ[10]:

1. Inisialisasi vektor masukan, inisialisasi laju pembelajaran α(0).
2. Selama kondisi berhenti tidak terpenuhi, lakukan langkah 3-7.
3. Untuk setiap pelatihan vektor masukan x, lakukan langkah 4-5.
4. Cari nilai J sehingga $\| x - w_j \|$bernilai minimum.
5. Perbaharui nilai wJ, dengan aturan sebagai berikut:
   Jika T = CJ, lakukan
   WJ(baru) = WJ(lama) + α[x - WJ(lama)];
   Jika T ≠ CJ, lakukan
   WJ(baru) = WJ(lama) - α[x - WJ(lama)];
6. Kurangi laju pembelajaran.
7. Uji kondisi berhenti:

Kondisi dapat berupa jumlah pasti iterasi yang dilakukan atau tingkat pembelajaran yang mencapai nilai tertentu.

Bobot hasil dari pelatihan pada metode LVQ ini akan disimpan pada file csv dan digunakan untuk proses pengujian. Proses pelatihan bobot ini ditujukan agar proses pengenalan pada objek dapat dilakukan dengan baik. Pada proses pelatihan, tingkat pembelajaran dan jumlah epoh atau jumlah banyaknya pelatihan dilakukan ini berpengaruh juga pada tingkat akurasi dari jaringan syaraf tiruan LVQ.

Pada pengujian LQV, fungsi aktivasi yang digunakan adalah fungsi aktivasi linear. Tujuan menggunakan fungsi

aktivasi linear adalah memperoleh keluaran yang sama dengan masukan, sesuai dengan rumus fungsi linear yaitu y = x. Fungsi aktivasi F1 akan memetakan y_in1 ke Y1 = 1 apabila |x – W1| < |x – W2|, dan y1 = 0 jika sebaliknya. Demikian pula fungsi aktivasi F2 akan memetakan y_in2 ke Y2 = 1 apabila |x – W2| < |x – W1|, dan y2 = 0 jika sebaliknya.

Setelah dilakukan pelatihan, maka akan diperoleh jaringan LVQ yang memiliki bobot-bobot antar neuron yang sudah mampu melakukan pengklasifikasian. Berikut adalah algoritma untuk pengujian pada metode LVQ:

1. Masukan data yang akan diuji beserta bobot yang telah dilatih.
2. Tentukan nilai J sehingga $\| x - w_j \|$ bernilai minimum.
3. Tentukan neuron dengan jarak minimum.
4. Objek akan dikenali dengan nilai minimum vektor masukan dengan kelas keluaran.

3. Hasil Percobaan dan Pembahasan

Pengujian dilakukan dengan memberikan masukan berupa citra ponsel sebanyak 250 citra yang terdiri dari 10 citra setiap kelasnya. Pengujian dilakukan dengan menggunakan bobot yang dilatih dengan beberapa perbedaan epoh dan tahap pembelajaran. Proses pengujian dilakukan dengan memproses setiap citra melalui *feature extraction* yang akan menghasilkan fitur vektor. Fitur vektor tersebut akan digunakan untuk proses pengenalan pada metode LVQ oleh bobot terlatih.

Tabel 1 Keterangan kelas pengujian

ID	Kelas
1	Samsung S9
2	Samsung S8
3	Samsung A8
4	Samsung J8
5	Samsung J7
6	iPhone X
7	iPhone 8+
8	iPhone 8
9	iPhone 7+
10	iPhone 7
11	Mi 6
12	Mi F1
13	Mi A2
14	Mi Max 3
15	Mi 8
16	Zenfone 6
17	Zenfone 5
18	Zenfone 4
19	Zenfone Max M1
20	Zenfone 3
21	Oppo F9
22	Oppo Find X
23	Oppo A3s
24	Oppo F7
25	Oppo A83

3.1 Hasil Pengujian Model Latih

Pengujian ditujukan untuk mencari bobot terlatih yang mampu mengenali setiap jenis ponsel dengan baik. Model latih dibedakan berdasarkan jumlah epoh dan nilai tahap pembelajaran. Perbedaan tersebut dapat menghasilkan bobot terlatih yang mampu mengenali jenis ponsel dengan baik. Pengujian dilakukan dengan menggunakan model latih 250 epoh dan tahap pembelajaran 0,002, model latih 250 epoh dan tahap pembelajaran 0,001, model latih 500 epoh dengan tahap pembelajaran 0,001, model latih 500 epoh dengan tahap pembelajaran 0,0001, model latih 750 epoh dengan tahap pembelajaran 0,00001. Berikut hasil dari pengujian terhadap model latih tersebut:

Tabel 2 Hasil pengujian model latih 250 epoh dan tahap pembelajaran 0,002

Kelas	Precision	Recall	F1-score
1	0.214	0.3	0.249
2	0.285	0.2	0.235
3	0	0	0
4	0	0	0
5	0	0	0
6	0.083	0.1	0.09
7	0	0	0
8	0	0	0
9	0	0	0
10	0	0	0
11	0.125	0.1	0.111
12	0.153	0.2	0.173
13	0.2	0.3	0.24
14	0.222	0.2	0.21
15	0.142	0.2	0.166
16	0.167	0.1	0.125
17	0	0	0
18	0	0	0
19	0.2	0.2	0.2
20	0	0	0
21	0.05	0.1	0.066
22	0.111	0.1	0.105
23	0	0	0
24	0	0	0
25	0.1	0.2	0.133
Average	0.082	0.092	0.084

Tabel 3 Hasil pengujian model latih 250 epoh dan tahap pembelajaran 0,001

Kelas	Precision	Recall	F1-score
1	0	0	0
2	0.153	0.2	0.173
3	0	0	0
4	0.1	0.1	0.1
5	0.125	0.1	0.11
6	0.833	0.1	0.018
7	0.833	0.1	0.018
8	0	0	0
9	0	0	0
10	0.074	0.2	0.108
11	0	0	0
12	0	0	0
13	0.2	0.1	0.133

Tabel 3 (Lanjutan) Hasil pengujian model latih 250 epoh dan tahap pembelajaran 0,001

14	0.125	0.1	0.11
15	0	0	0
16	0	0	0
17	0.285	0.2	0.235
18	0.1	0.1	0.1
19	0	0	0
20	0.25	0.2	0.22
21	0	0	0
22	0	0	0
23	0.444	0.4	0.42
24	0.153	0.2	0.173
25	0.285	0.2	0.235
Average	0.1584	0.092	0.08612

Tabel 4 Hasil pengujian model latih 500 epoh dan tahap pembelajaran 0,001

Kelas	**Precision**	**Recall**	**F1-score**
1	0	0	0
2	0	0	0
3	0.111	0.1	0.105
4	0	0	0
5	0	0	0
6	0.133	0.2	0.159
7	0	0	0
8	0.142	0.2	0.166
9	0	0	0
10	0.062	0.1	0.076
11	0	0	0
12	0.1	0.1	0.1
13	0.142	0.1	0.117
14	0.055	0.1	0.07
15	0.667	0.2	0.307
16	0	0	0
17	0	0	0
18	0	0	0
19	0	0	0
20	0	0	0
21	0.093	0.3	0.141
22	0	0	0
23	1	0.3	0.461
24	0.071	0.1	0.083
25	0.167	0.1	0.125
Average	0.109	0.076	0.076

Tabel 5 Hasil pengujian model latih 500 epoh dan tahap pembelajaran 0,0001

Kelas	**Precision**	**Recall**	**F1-score**
1	0.142	0.2	0.166
2	0.083	0.1	0.09
3	0.1	0.1	0.1
4	0.1	0.1	0.1
5	0.09	0.1	0.095
6	0	0	0
7	0.4	0.2	0.266
8	0	0	0
9	0.083	0.1	0.09
10	0.052	0.1	0.068
11	0.142	0.3	0.193
12	0	0	0
13	0.153	0.2	0.173
14	0.125	0.1	0.111
15	0	0	0

Tabel 5 (Lanjutan) Hasil pengujian model latih 500 epoh dan tahap pembelajaran 0,0001

16	0.09	0.1	0.095
17	0.166	0.3	0.214
18	0.142	0.1	0.117
19	1	0.1	0.181
20	0	0	0
21	0	0	0
22	0.4	0.2	0.266
23	0	0	0
24	0	0	0
25	0	0	0
Average	0.131	0.096	0.093

Tabel 6 Hasil pengujian model latih 750 epoh dan tahap pembelajaran 0,00001

Kelas	**Precision**	**Recall**	**F1-score**
1	0.167	0.1	0.125
2	0.167	0.1	0.125
3	0.222	0.4	0.285
4	0.125	2	0.235
5	0	0	0
6	0.167	0.1	0.125
7	0	0	0
8	0	0	0
9	0	0	0
10	0	0	0
11	0	0	0
12	0.181	0.2	0.19
13	0.058	0.1	0.073
14	0.09	0.1	0.094
15	0.333	0.1	0.153
16	0.047	0.1	0.063
17	0.105	0.2	0.137
18	0.222	0.2	0.21
19	0	0	0
20	0	0	0
21	0	0	0
22	0	0	0
23	0	0	0
24	0	0	0
25	0	0	0
Average	0.075	0.148	0.0728

Dari hasil pengujian yang telah dilakukan, didapatkan hasil pengujian sebagai berikut:

1. Penambahan jumlah epoh dan pengecilan nilai tahap pembelajaran tidak selalu menghasilkan bobot terlatih terbaik. Hal ini dapat dilihat pada model latih 500 epoh dengan tahap pembelajaran 0,0001 menghasilkan nilai *precision*, *recall*, dan f1-score lebih baik disbanding model latih 750 epoh dengan tahap pembelajaran 0,00001.
2. Hasil terbaik yang didapatkan dari model latih yaitu model latih dengan 500 epoh dan tahap pembelajaran 0,0001. Model latih mampu menghasilkan nilai rata-rata *precision* sebesar 0,131, *recall* sebesar 0,096, dan f1-score 0,093.
3. Model latih 500 epoh dengan tahap pembelajaran 0,0001 mampu mengenali sebanyak 16 jenis ponsel dari total 25 jenis. Hal

tersebut merupakan nilai tertinggi dari model latih lainnya.

4. Kesimpulan dan Saran

Kesimpulan yang didapat dari hasil pengujian yang telah dilakukan yaitu:

1. Hasil dari pengenalan oleh program aplikasi memiliki akurasi yang kurang baik.
2. Fitur yang diambil harus memiliki karakteristik yang kuat atau detail sehingga dapat digunakan untuk membedakan jenis-jenis ponsel.
3. Fitur bentuk, warna, dan tekstur tidak mampu menghasilkan pengenalan terhadap jenis ponsel dengan baik.

Beberapa saran yang dapat diberikan untuk upaya pengembangan program aplikasi lebih lanjut adalah:

1. Penambahan jumlah data gambar setiap kelas untuk proses pelatihan dan pengujian.
2. Program pengenalan dapat mengenali semua jenis ponsel.
3. Pengambilan fitur berupa detail dari ponsel agar memiliki fitur dengan karakteristik yang kuat.
4. Jumlah produk ponsel yang ditampilkan lebih banyak.

REFERENSI

[1] Rebecca Sentence, The new wave of visual search: what it can do, and what might be possible, https://searchenginewatch.com/2016/10/13/the-new-wave-of-visual-search-what-it-can-do-and-what-might-be-possible/, 27 Agustus 2018.

[2] Ying Li, C.C. Jay Kuo, and X. Wan, Image Databases: Search and Retrieval of Digital Imagery, (Hoboken: John Wiley & Sons, Inc., 2002), h.262.

[3] Digital Image, Computer Sciences, https://www.encyclopedia.com/computing/news-wires-white-papers-and-books/digital-images, 4 September 2018.

[4] Margaret Rouse, Database, https://searchsqlserver.techtarget.com/definition/database, 29 September 2018.

[5] Codecademy, What is REST, https://www.codecademy.com/articles/what-is-rest, 1 Oktober 2018.

[6] Margaret Rouse, JSON (JavaScript Object Notation), https://searchwindevelopment.techtarget.com/definition/JSON-Javascript-Object-Notation, 1 Oktober 2018.

[7] K. Kranti Kumar dan T. Venu Gopal, CBIR: Content Based Image Retrieval, https://www.researchgate.net/publication/235634738 , 1 September 2018.

[8] Satya Mallick, Histogram of Oriented Gradients, https://www.learnopencv.com/histogram-of-oriented-gradients/, 30 Agustus 2018.

[9] Perfect Makanju, K Means and Image Quantization [Part 2], https://medium.com/consonance/k-means-and-image-quantization-part-2-be0a62c50c11, 10 September 2018.

[10] Laurene Fausett, Fundamental of Neural Networks Architectures, Algorithms, and Aplication, (Upper Saddle River: Prenctice Hall, 1994), h.190.

Nickolas Cornelius Siantar, merupakan mahasiswa program Sarjana S1, program studi Teknik Informatika, Fakultas Teknologi Informasi, Universitas Tarumanagara.