SELEKSI FITUR *INFORMATION GAIN* PADA KLASIFIKASI CITRA MAKANAN MENGGUNAKAN *HUE SATURATION VALUE* DAN *GRAY LEVEL CO-OCCURRENCE MATRIX*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:

Frisma Yessy Nabella

NIM: 155150201111062

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG

2018

PENGESAHAN

SELEKSI FITUR *INFORMATION GAIN* PADA KLASIFIKASI CITRA MAKANAN MENGGUNAKAN *HUE SATURATION VALUE* DAN *GRAY LEVEL CO-OCCURRENCE MATRIX*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Frisma Yessy Nabella
NIM: 155150201111062

Skripsi ini telah diuji dan dinyatakan lulus pada
27 Desember 2018
Telah diperiksa dan disetujui oleh:

Pembimbing I	Pembimbing II
Yuita Arum Sari S.Kom., M.Kom	Randy Cahya Wihandika S.ST., M.Kom
NIK: 201609 880715 2 001	NIK: 201405 880206 1 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T., M.T., Ph.D
NIP. 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 27 Desember 2018

Frisma Yessy Nabella

NIM: 155150201111062

PRAKATA

Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat, taufik dan hidayah-Nya sehingga laporan skripsi berjudul "Seleksi Fitur *Information Gain* Pada Klasifikasi Citra Makanan Menggunakan *Hue Saturation Value* Dan *Gray Level Co-Occurrence Matrix*" ini dapat terselesaikan.

Penulis menyadari bahwa skripsi ini tidak akan berhasil tanpa bantuan dari beberapa pihak. Oleh karena itu, penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Bapak Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya.
2. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
3. Bapak Agus Wahyu Widodo, S.T, M.Cs selaku Ketua Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
4. Ibu Yuita Arum Sari, S.Kom, M.Kom selaku dosen pembimbing I yang telah membagikan ilmu serta arahan dalam proses pengerjaan skripsi ini.
5. Bapak Randy Cahya Wihandika, S.ST., M.Kom selaku dosen pembimbing II yang telah membagikan ilmu serta arahan dalam proses pengerjaan skripsi ini.
6. Bapak Imam Colissodin, S.Si., M.Kom selaku dosen pembimbing akademik yang telah memberikan kritik dan saran selama proses perkuliahan.
7. Kedua orang tua penulis, Ibu Nurmawati dan Bapak Subandianto serta adik penulis, Anindya yang senantiasa memberikan do'a, semangat dan kasih sayang kepada penulis.
8. Sahabat penulis, Nadya, Enggar, Feni, Meli, Diva, Farhan, kerdus squad, pejuang S.Kom squad, teman-teman K-RISMA, teman-teman IKMBM dan sahabat lainnya yang tidak bisa disebutkan satu persatu.
9. Teman-teman seperjuangan penulis khususnya Caca, Devin, Karunia, Fauzi, Aan, Sulaiman, Dyva dan teman-teman induksi riset kelas F.
10. Seluruh civitas academika Informatika Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi di Informatika Universitas Brawijaya dan selama penyelesaian skripsi ini.
11. Semua pihak yang tidak dapat dituliskan satu persatu yang terlibat dalam membantu penulis menyelesaikan skripsi ini.

Malang, 27 Desember 2018

Penulis

frismanabella@student.ub.ac.id

ABSTRAK

Frisma Yessy Nabella, Seleksi Fitur Information Gain Pada Klasifikasi Citra Makanan Menggunakan *Hue Saturaion Value* Dan *Gray Level Co-Occurrence Matrix*

Pembimbing: Yuita Arum Sari, S.Kom., M.Kom dan Randy Cahya Wihandika, S.ST., M.Kom.

Pemilihan makanan menjadi masalah terutama bagi mereka yang baru pertama kali mencicipi atau mencari suatu makanan yang belum diketahui bahan dan kadar gizinya. Klasifikasi penting dilakukan untuk menilai diet pada penderita diabetes dan penderita alergi makanan serta membantu dalam menemukan kalori makanan, nilai nutrisi dan sebagai referensi makanan. Pemilihan makanan juga diperlukan bagi para penyandang tunanetra untuk mengetahui nama suatu makanan. Mengonsumsi makanan yang benar juga penting dilakukan untuk menghindari penyakit dan menjalani gaya hidup sehat. Klasifikasi makanan dapat dilakukan menggunakan metode-metode yang ada pada *computer vision*. Proses klasifikasi diawali dengan *pre-processing* citra yang menghasilkan citra tersegmentasi. Hasil dari *pre-processing* kemudian masuk ke tahap ekstraksi fitur. Penelitian ini menggunakan eksraksi warna *Hue Saturation Value* (HSV) dan tekstur *Gray Level Co-Occurrence Matrix* (GLCM). Selanjutnya dilakukan seleksi fitur berdasarkan fitur-fitur yang telah didapatkan untuk mengambil fitur-fitur yang relevan saja guna meningkatkan akurasi dan mengurangi beban kerja sistem menggunakan seleksi fitur *Information Gain*. Proses yang terakhir yaitu klasifikasi. Penelitian ini menggunakan metode klasifikasi *K-Nearest Neighbor*. Hasil akurasi terbaik pada penelitian ini yaitu 95,24% dengan hanya menggunakan HSV menggunakan seleksi fitur dengan nilai k=1. Pada kombinasi HSV dan GLCM dengan *Information Gain* mampu meningkatkan akurasi dari 57,14% menjadi 87,61%. Begitu juga dengan fitur GLCM saja menggunakan *Information Gain* yang dapat meningkatkan akurasi dari 57,14% menjadi 74,28%. Dengan demikian, seleksi fitur *Information Gain* sangat berperan dalam meningkatkan akurasi dan mampu mendapatkan fitur-fitur relevan apabila menggunakan fitur berjumlah banyak. Sedangkan apabila jumlah fitur sedikit, peningkatan akurasi tidak terlalu signifikan namun tetap mampu mengurangi beban kerja sistem.

Kata kunci: klasifikasi, makanan, HSV, GLCM, information gain

ABSTRACT

The choice of food is a problem, especially for those who are the first time tasting or looking for a food that has no known ingredients and nutritional content. An important classification is done to assess diets in diabetics and food allergies and help in finding food calories, nutritional values and food references. The choice of food is also needed for blind people to know the name of a food. Eating the right foods is also important to avoid disease and lead a healthy lifestyle. Food classification can be done using methods in computer vision. The classification process begins with pre-processing images that produce segmented images. The results of pre-processing then enter the feature extraction stage. This study uses the extraction of Hue Saturation Value (HSV) and Gray Level Co-Occurrence Matrix (GLCM) textures. Furthermore, feature selection is based on the features that have been obtained to take only relevant features to improve accuracy and reduce system workload using the Information Gain feature selection. The last process is classification. This study uses the K-Nearest Neighbor classification method. The best accuracy results in this study were 95.24% using only HSV using feature selection with a value of k = 1. The combination of HSV and GLCM with Information Gain was able to increase accuracy from 57.14% to 87.61%. Likewise, the GLCM feature only uses Information Gain which can increase accuracy from 57.14% to 74.28%. Thus, the Information Gain feature selection plays an important role in improving accuracy and being able to get relevant features when using multiple features. Whereas if the number of features is small, the accuracy increase is not too significant but still able to reduce the system workload.

Keywords: classification, food, HSV, GLCM, information gain

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

Pada bab ini, dijelaskan latar belakang, rumusan masalah, tujuan, manfaat, batasan masalah dan sistematika penulisan penelitian.

1.1 Latar belakang

Memilih makanan merupakan hal penting bagi penderita penyakit tertentu. Contohnya pada penderita jantung *coroner* yang harus memperhatikan pemenuhan gizi yang tepat agar tidak memicu keaadaan jantung menjadi lebih buruk (Puspitasari, Ratnawati, & Fauzi, 2018). Dalam penerapan diet juga dibutuhkan pengaturan menu makan yang tepat guna menghindari dan membatasi makanan yang menyebabkan peningkatan kolesterol dan tekanan (Purwati, 2002). Pemilihan makanan bagi sebagian orang mungkin bukan suatu masalah bagi mereka yang telah mengenali makanan tersebut. Namun berbeda bagi sebagian orang yang baru pertama kali ingin mencicipi atau mencari suatu makanan yang belum pernah diketahui bahan dan kadar gizi dari makanan (Joutou & Yanai, 2009). Para penyandang tunanetra juga mengalami kesulitan dalam pemilihan makanan. Mereka tidak dapat melihat harga unit, bahan dan nutrisi dari makanan. Dengan demikian para penderita tunanetra membutuhkan suatu sistem untuk mengetahui nama suatu makanan (Hoonlor, et al., 2015).

Mengonsumsi makanan yang benar dapat dilakukan untuk menghindari penyakit dan menjalani gaya hidup sehat. Oleh sebab itu, penting untuk melakukan klasifikasi makanan. Makanan yang sangat bermacam-macam menjadi tantangan dalam pengklasifikasian makanan. Klasifikasi makanan juga merupakan hal yang penting untuk menilai diet pada penderita diabetes dan orang yang menderita alergi makanan serta membantu dalam menemukan kalori makanan, nilai nutrisi dan sebagai referensi makanan. Klasifikasi makanan dapat dilakukan menggunakan metode-metode yang ada pada *computer vision* (Jahan, Kekha, & Quadri, 2018).

Penelitian He, et al. (2014) telah mencoba mengatasi permasalahan tersebut dengan melakukan klasifikasi 42 kelas makanan. Metode yang digunakan yaitu ekstraksi ciri *Dominant Color Descriptor* (DCD), *Multi-scale Dense SIFT* (MDSIFT), *Scalable Color Descriptor* (SCD), *Entropy-Based Categorization and Fractal Dimension* (EFD), *Gabor-Based Image Decomposition and Fractal Dimension Estimation* (GFD) dan *Scale Invariant Feature Transforms* (SIFT) menggunakan metode klasifikasi KNN. Hasil dari penelitian ini didapatkan bahwa kombinasi dari fitur DCD, MDSIFT dan SCD menghasilkan akurasi yang terbaik yaitu 85,1% dimana lebih baik 22% dibandingkan kombinasi fitur lainnya.

Penelitian Mohanaiah, Sathyanarayana, dan GuruKumar (2013) mengatakan terdapat beberapa cara melakukan ekstraksi fitur sepeti melalui warna, tekstur, dan bentuk. GLCM merupakan salah satu algoritme ekstraksi tekstur populer yang telah terbukti baik untuk mengekstraksi tekstur dan mendapatkan hasil akurasi yang tinggi. Penelitian lain Hastuti, Dewi, & Widodo

(2017) menggunakan GLCM dengan sudut 0˚, 45˚, 90˚, dan 135˚ *menggunakan K-NN* mendapatkan hasil klasikasi terbaik menggunakan kombinasi fitur sudut empat arah atau 0˚ dan HSV yaitu dengan akurasi 100% dan nilai *k*=3, *k*= 11 atau *k*=15.

Penelitian Alhassan & Khader (2014) menggunakan fitur warna yang diekstraksi oleh *Color Moment* pada ruang warna HSV menunjukkan bahwa penggunakan fitur warna dan tekstur mendapatkan hasil yang lebih akurat daripada hanya menggunakan salah satunya saja. Penelitian (Kusumanto, Pambudi, & Tomponu, 2011) menyatakan bahwa penggunaan ruang warna HSV dapat mendeteksi dan mengurangi pengaruh intensitas cahaya dari luar. Penelitian (Singh & Hemachandran, 2012) menyebutkan bahwa RGB benar-benar menggambarkan warna yang terlihat sehingga tidak menyediakan titik awal yang berguna untuk mewakili warna gambar dan juga ruang warna RGB tidak memiliki perseptual yang seragam. Maka dibutuhkan untuk menghasilkan ruang warna lain contohnya HSV, CIELAB dan CIELUV. Hasil terbaik dari ketiga ruang warna tersebut yaitu HSV karena memiliki keseragaman perseptual. Ruang warna HSV banyak digunakan dalam bidang penglihatan warna dimana setiap channelnya berhubungan erat dengan kategori persepsi warna manusia.

Hasil dari ekstraksi fitur sering kali mendapatkan fitur terlalu banyak yang menyebabkan terdapat beban kerja tinggi sehingga dapat menurunkan akurasi. Dengan demikian dibutuhkan untuk menghilangkan fitur-fitur tersebut untuk mendapatkan fitur yang relevan dan tidak berlebihan yang dapat memberikan solusi optimal tanpa mengurangi akurasi klasifikasi (Mwadulo, 2016). *Information gain* merupakan metode seleksi fitur yang bekerja dengan cara melakukan pemeringkatan secara sederhana (Chormunge & Jena, 2016). Penelitian oleh Shaltout, et al. (2014) disebutkan bahwa *information gain* mampu mendeteksi fitur-fitur yang paling relevan dengan menghitung nilai *entropy* berdasarkan kelas-kelas tertentu.

Terdapat banyak metode klasifikasi yang dapat digunakan dalam mengklasifiksikan suatu objek. Penelitian Redjeki (2013) membandingkan akurasi dari dua metode yaitu KNN dan *Backpropagation*. Hasil penelitian tersebut meyimpulkan bahwa algoritme KNN mendapatkan hasil identifikasi yang lebih baik daripada algoritme *Backpropagation*. Penelitian lain oleh Priambodo, et al. (2015) melakukan klasifikasi penyakit tanaman jeruk keprok menggunakan metode KNN dengan hasil akurasi 96,67% membuktikan bahwa metode KNN baik dalam mengklasifikasikan citra.

Dilatarbelakangi dengan penjelasan diatas maka penulis mengusulkan untuk melakukan penelitian dengan topik "Seleksi Fitur *Information Gain* Pada Klasifikasi Citra Makanan Menggunakan *Hue Saturation Value* dan *Gray Level Cooccurence Matrix*". Diharapkan penelitian ini mampu mendapatkan fitur relevan yang didapatkan dari ekstraksi warna HSV dan tekstur GLCM sehingga mampu mengklasifikasikan citra makanan dengan baik serta mampu mengurangi beban kerja berdasarkan fitur-fitur yang didapatkan dari hasil seleksi fitur.

1.2 Rumusan masalah

1. Bagaimana pengaruh ekstraksi warna HSV dan tekstur GLCM pada permasalahan klasifikasi citra makanan?
2. Bagaimana pengaruh seleksi fitur *Information Gain* berdasarkan ekstraksi warna HSV dan tekstur GLCM pada permasalahan klasifikasi citra makanan?
3. Bagaimana hasil akurasi dari dari seleksi fitur *Information Gain* pada permasalahan klasifikasi citra makanan menggunakan ekstraksi warna HSV dan tekstur GLCM?

1.3 Tujuan

Tujuan dilakukan penelitian ini adalah sebagai berikut.

1. Mengetahui pengaruh ekstraksi warna HSV dan tekstur GLCM untuk menyelesaikan permasalahan klasifikasi citra makanan.
2. Mengetahui pengaruh seleksi fitur *Information Gain* berdasarkan ekstraksi warna HSV dan tekstur GLCM untuk menyelesaikan permasalahan klasifikasi citra makanan.
3. Mengetahui hasil akurasi dari dari seleksi fitur *Information* Gain pada permasalahan klasifikasi citra makanan menggunakan ekstraksi warna HSV dan tekstur GLCM.

1.4 Manfaat

Penelitian ini diharapkan dapat memberikan beberapa manfaat sebagai berikut.

1. Melalui penelitian ini dapat diketahui pengaruh dari ekstraksi warna HSV dan tekstur GLCM untuk menyelesaikan permasalahan klasifikasi citra makanan yang dapat dijadikan referensi untuk melakukan penelitian lain yang berkaitan dengan algoritme yang digunakan.
2. Melalui penelitian ini dapat diketahui pengaruh seleksi fitur *Information Gain* berdasarkan ekstraksi warna HSV dan tekstur GLCM untuk menyelesaikan permasalahan klasifikasi citra makanan yang dapat dijadikan referensi untuk melakukan penelitian lain yang berkaitan dengan algoritme yang digunakan.
3. Melalui penelitian ini dapat diketahui hasil akurasi dari dari seleksi fitur *Information* Gain pada permasalahan klasifikasi citra makanan menggunakan ekstraksi warna HSV dan tekstur *GLCM* sehingga dapat menjadi bahan pertimbangan untuk melakukan penelitian lain yang berkaitan dengan algoritme yang digunakan.

1.5 Batasan masalah

Dalam penelitian terdapat beberapa batasan masalah yang dipaparkan sebagai berikut.

1. Kelas makanan berjumlah dua puluh satu kelas meliputi Donat, Roti Gandum, Roti Tawar, Telor, *Fried Chicken*, Rendang, Mentimun, Selada, Tomat, *Strawberry*, Pisang Hijau, Pisang Kuning, Jeruk, Nasi Merah, Oreo, Beng-Beng, Soba Mie, Gerry Salut, Biskuat, Milo Nuggets, Genji Pie.
2. Setiap kelasnya terdiri dari 10 citra cata latih dan 5 citra data uji.
3. Citra makanan data latih maupun data uji berupa citra makanan tunggal.
4. *Color Moment* pada ekstraksi warna menggunakan ruang warna HSV.
5. Ekstraksi fitur tekstur GLCM pada menggunakan empat sudut, yaitu sudut 0˚, 45˚, 90˚, dan 135˚.

1.6 Sistematika pembahasan

Bagian menjelaskan struktur penelitian ini dimulai dari Bab Pendahuluan sampai Bab Penutup secara singkat. Diharapkan sub bab ini dapat membantu pembaca dalam memahami isi penelitian.

BAB I Pendahuluan

Menguraikan latar belakang dari penelitian, perumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelsitian, dan sistematika pembahasan.

BAB II Landasan Kepustakaan

Menguraikan dasar teori yang berkaitan dengan makanan, ekstraksi warna HSV, ekstraksi GLCM, seleksi fitur *information gain* dan metode klasifikasi *KNN* serta penelitian terdahulu yang menjadi referensi dan acuan pada penelitian ini.

BAB III Metodologi Penelitian

Metode-metode dan tahapan yang digunakan dalam menyelesaikan penelitian seleksi fitur information gain untuk klasifikasi citra makanan berdasarkan HSV dan GLCM.

BAB IV Perancangan

Berisi perancangan dan penjelasannya yang meliputi perancangan algoritme, perhitungan manual, serta skenario pengujian.

BAB V Implementasi

Berisi tentang pengimplementasian dan pembahasan sistem information gain pada klasifikasi citra makanan menggunakan metode KNN, batasan-batasan implementasi serta metode-metode yang digunakan didalamnya.

BAB VI Pengujian dan Analisis

Bab pengujian dan analisis berisi hasil pengujian dan analisis dari implementasi yang telah dilakukan dengan melakukan perhitungan akurasi sistem.

BAB VII Penutup

Bab penutup membahas mengenai kesimpulan yang diperoleh dari pengujian yang dilakukan serta saran-saran untuk pengembangan penelitian lebih lanjut.

BAB 2 LANDASAN KEPUSTAKAAN

Landasan kepustakaan berisi pembahasan mengenai penelitian-penelitian terdahulu yang bekaitan dengan permasalahan pada penelitian ini. Pada landasan kepustakaan juga diuraikan teori, konsep, metode pendukung yang digunakan di penelitian ini seperti *pre-processing*, ekstraksi warna HSV, ekstraksi tekstur GLCM, seleksi fitur *Information Gain*, dan metode *K – Nearest Neighbor*.

2.1. Kajian Pustaka

Kajian pustaka pada penelitian ini membahas tentang penelitian-penelitian yang sudah dilakukan sebelumnya dan perbedaan dengan penelitian yang akan dilakukan. Penelitian-penelitian ini menunjang penelitian penulis dengan memaparkan kajian mengenai metode yang digunakan. Terdapat empat penelitian yang berhubungan dengan penelitian ini dimana sama-sama menggunakan teknologi pengolahan citra dalam mengklasifikasikan citra yang diantaranya juga menggunakan seleksi fitur *Information Gain*, metode GLCM, HSV dan KNN untuk mengklasifikasikan citra. Perbedaan penelitian sebelumnya dan penelitian ini dapat dilihat pada Tabel 2.1.

Table 0.1 Kajian Pustaka

No.	Referensi	Persamaan	Referensi	Perbedaan	
				Penelitian ini	Hasil Referensi
1.	Hastuti, Dewi, & Widodo (2017). Identifikasi Kondisi Kesehatan Ayam Petelur Berdasarkan Ciri Warna HSV Dan Gray Level Cooccurence Matrix (GLCM) Pada Citra Jengger Dengan Klasifikasi K-Nearest Neighbour	- Menggun a-kan Color Moment HSV untuk ekstraksi warna - Menggun a- kan GLCM untuk ekstraksi tekstur dengan sudut 0˚, 45˚, 90˚, dan 135˚ - Menggun	-Fitur HSV berupa kadar hsv setiap *channel* -Fitur GLCM yang digunakan yaitu entropi, energi, homogenitas, kontras -Tidak menggunakan seleksi fitur -Objek yang digunakan yaitu citra	-Fitur HSV berupa *Color Moment* meliputi *mean*, deviasi standar, *skewness* dan *kurtosis*. -Fitur GLCM yang digunakan yaitu *mean X, mean Y, variance X, variance Y, energy, entropy, contrast,*	-Menggunakan 24 fitur total. -Akurasi terbaik didapat pada saat klasifikasi dengan GLCM 4 Arah atau sudut 0˚ + HSV dan jumlah K=3, K=11, dan K=15 yaitu 100%.

		a kan metode klasifikasi KNN	jengger ayam	*dissimilarity*, *homogeneity dan correlation* -Mengguna kan seleksi fitur *Information Gain.* -Objek yang digukan yaitu citra makanan.	
2.	Priambodo, at al. (2015). Implementasi metode k-nearest neighbour untuk identifikasi penyakit tanaman jeruk kapok berdasarkan citra daun.	-Objek penelitian berupa citra -Menggu-nakan metode klasifikasi KNN	-Objek yang digunakan yaitu citra daun jeruk -Fitur yang digunakan hanya fitur warna. -Fitur warna dihitung melalui ruang warna RGB dan dicari nilai rata-rata setiap channelnya.	-Objek yang digunakan yaitu citra makanan -Fitur yang digunakan yaitu warna dan tekstur. -Fitur warna dihitung melalui ruang warna HSV denggan menggunak an Color Moment berupa mean, deviasi standar, skewness dan kurtosis.	-akurasi terbaik mencapai 96,67%.
3.	Aini, Sari, & Arwan (2018) Seleksi Fitur Information Gain untuk	-Menggun a kan Informati on Gain	-Objek penelitian berupa teks.	-Objek penelitian berupa citra.	-Hasil akurasi sebesar 92,31% menggunakan 6 fitur dengan nilai K=25 dan

	Klasifikasi Penyakit Jantung Menggunakan Kombinasi Metode K-Nearest Neighbor dan Naïve Bayes	untuk melakukan seleksi fitur. - Menggunakan metode Klasifikasi KNN.			pada saat pengujian sebaran kelas tidak seimbang menggunakan 4 fitur dengan nilai K=35.
4.	He, et al. (2014) Analysis OF FOOD IMAGES: FEATURES AND CLASSIFICATION	-Melakukan klasifikasi citra makanan -Menggunakan metode klasifikasi KNN	-Ekstraksi fitur yang digunakan yaitu kombinasi antara DCD, MDSIFT, SCD, EFD, GFD dan SIFT -Tidak menggunakan seleksi fitur.	-Ekstraksi fitur yang digunakan yaitu HSV dan GLCM -Menggu-nakan seleksi fitur Information Gain.	-akurasi tertinggi yang dihasilkan yaitu 85,1% dengan menggunakan kombinasi fitur DCD+MDSIFT+SCD kombinasi ketiga fitur tersebut mampu meningkatkan hasil klasifikasi makanan sekitar 22%

Penelitian Hastuti, Dewi, & Widodo (2017) bertujuan mengklasifikasikan citra jengger ayam menjadi dua kelas yaitu sehat dan sakit. Penelitian tersebut menggunakan *Color Moment* HSV untuk mendapatkan fitur warna dan GLCM untuk mendapatkan fitur tekstur dengan mengambil fitur statistik entropi, energi, homogenitas, kontras dan korelasi dan menggunakan metode KNN untuk menentukan kelas berdasarkan fitur-fitur yang telah didapatkan. Hasil dari penelitian tersebut mendapatkan akurasi terbaik 100% dengan GLCM 4 arah atau kombinasi GLCM sudut 0˚ ditambah *Color Moment* HSV dan jumlah ketetanggaan yaitu K=3, K=11, K=15

Penelitian Priambodo, at al. (2015) yang mencoba mengklasifiksikan penyakit tanaman jeruk keprok menggunakan metode KNN dengan hasil akurasi 96,67% membuktikan bahwa metode KNN baik dalam mengklasifikasikan citra. Pada penelitian lain oleh (Redjeki, 2013) yang mencoba membandingkan akurasi dari dua metode yaitu KNN dan Jaringan *Syaraf* Tiruan yaitu *Backpropagation* dalam mengidentifikasi mendapatkan hasil bahwa algoritme KNN mendapatkan akurasi sebesar 100% dengan nilai k=3, k=7, dan k=9 dan mendapatkan akurasi sebesar 90% dengan nilai learning rate 0,1 dan *error* sebesar 0,001. Dari hasil penelitian tersebut maka disimpulkan bahwa algoritme KNN memberikan hasil identifikasi lebih baik daripada algoritme *Backpropagation*.

Penelitian Aini, Sari, & Arwan (2018) mengatakan bahwa seleksi fitur *Information Gain* baik diterapkan untuk klasifikasi, dengan akurasi yang didapatkan yaitu 92,31% membuktikan bahwa fitur-fitur yang terpilih dengan menggunakan information gain relevan dengan penelitian yang dilakukan, fitur yang didapatkan dengan nilai K=35 yaitu berjumlah 4 fitur yaitu thal, jenis nyeri dada, flourosopy dan rata-rata detak jantung mampu mewakili 13 fitur yang ada sebelum dilakukan seleksi fitur.

Penelitian He, et al. (2014) yaitu mengklasifikasikan 42 kelas citra makanan dengan menggunakan ekstraksi fitur, warna, tekstur dan daerah lokal, metode-metode yang digunakan yaitu DCD, MDSIFT, SCD, EFD, GFD dan SIFT dengan menggunakan metode klasifikasi KNN mendapatkan akurasi tertinggi yang dihasilkan yaitu 85,1% dengan menggunakan kombinasi fitur DCD+MDSIFT+SCD kombinasi ketiga fitur tersebut mampu meningkatkan hasil klasifikasi makanan sekitar 22%.

Berdasarkan kajian pustaka di atas, dapat disimpulkan bahwa ekstraksi fitur warna HSV baik digunakan karena mampu memisahkan komponen intensitas citra warna. Begitu juga dengan GLCM yang merupakan metode ekstraksi fitur yang cukup mumpuni serta metode klasifikasi KNN yang mampu digunakan untuk mengklasifikasikan citra berdasarkan fitur-fitur yang telah didapatkan dengan membandingkan seberapa mirip fitur-fitur tersebut metode ini juga membuktikan dapat mempunyai nilai akurasi yang tinggi pada pengklasifikasian citra. Serta *Information Gain* yang mempu melakukan seleksi fitur dan mendapatkan fitur-fitur yang relevan dengan penelitian. Sehingga disimpulkan bahwa metode-metode ini baik digunakan untuk melakukan klasifikasi citra makanan.

2.2. Segmentasi

Segmentasi dalam pemrosesan citra bertujuan untuk membagi citra menjadi beberapa daerah yang memiliki kemiripan pada atributnya. Citra yang hanya memiliki satu objek, akan dibedakan antara objek dengan latar belakangnya. Segmentasi juga bisa dilakukan sebagai langkah pertama untuk pengklasifikasian objek. Teknik dalam segmentasi didasarkan oleh nilai keabuan ketidaksinambungan dan kesamaan antar pikselnya (Kadir & Susanto, 2013). Penerapan segmentasi dapat dilakukan dengan memanfaatkan teknik-teknik pada citra digital, beberapa diantaranya yaitu: *filtering, thresholding*, dan *morphologicall image*.

2.2.1. Filtering

Fitering digunakan untuk menghilangkan noise pada citra, atau untuk meningkatkan detail suatu citra. Salah satu metode *filtering* yaitu *Gaussian filtering.* Fungsi *Gaussian* memiliki aplikasi penting pada banyak bidang matematika termasuk pengolahan citra. Karakteristik yang membuat fungsi *Gaussian* baik digunakan untuk *filtering* yaitu dapat merapikan gambar atau mendeteksi tepi setelah melakukan filtering (Shapiro & Stockman, 2000).

2.2.2. Ruang Warna L*a*b*

Ruang warna L*a*b* bertujuan agar dapat menyelaraskan warna menjadi lebih uniform melalui mata normal manusia, dimana apabila terjadi perubahan nilai warna maka perubahan secara visual dapat menghasilkan perubahan porsi yang sama. Ruang warna l*a*b* merepresentasikan semua warna yang dapat dilihat manusia. Kelebihan dari ruang warna l*a*b* yaitu tidak bergantung dengan peralatan yang digunakan. Terdapat tiga channel pada ruang warna l*a*b yaitu l* yang merupakan luminance dan dua komponen warna a* dan b* dimana keduanya menunjukkan intensitas tiap warna (Madenda, 2015).

2.2.3. Thresholding

Thresholding adalah salah satu metode segmentasi citra untuk memisahkan objek dan latar belakang (*background*) berdasarkan gelap terang atau hitam putih dalam suatu citra. Untuk memperoleh nilai *thresholding* perlu adanya perhitungan. Langkah awal untuk memperoleh nilai *thresholding* dengan cara membuat histogram, yakni untuk mengetahui tingkat keabuan pada jumlah piksel objek (Angraini, et al. 2015). Salah satu teknik yang dapat digunakan untuk melakukan *thresholding* adalah Otsu *thresholding*.

2.2.4. Operasi *Closing*

Operasi *closing* didahului dengan melakukan operasi dilasi lalu diikuti dengan operasi erosi (Kadir & Susanto, 2012). Dilasi merupakan proses morfologi yang bertujuan untuk meluaskan ukuran dari suatu objek. Sedangkan erosi merupakan proses morfologi yang berfungsi untuk mengecilkan ukuran dari suatu objek (Madenda, 2015).

2.3. Ekstraksi Fitur

Ekstraksi fitur adalah proses untuk pengambilan ciri dari suatu bentuk objek dalam citra. Melalui ciri bentuk objek tersebut dapat diambil nilai yang dapat digunakan untuk proses lebih lanjut. Pengambilan nilai ekstraksi fitur dengan cara menghitung jumlah piksel berdasarkan arah penelusuran pada koordinat vertical, horizontal, dan diagonal kiri maupun kanan.

Menurut Pamungkas (2015), berikut merupakan beberapa macam dari ekstraksi fitur Citra :

1. Ekstrasi fitur berdasarkan bentuk

Ekstrasi fitur berdasarkan bentuk digunakan untuk membedakan antara objek satu dengan yang lain. Biasanya ekstraksi fitur berdasarkan bentuk mempunyai nilai antara 0 sampai 1.

2. Ekstraksi fitur berdasarkan tekstur

Ekstraksi tekstur digunakan untuk membedakan objek satu dengan lainnya dengan cara menggunakan ciri orde statistik. Ciri orde statistik dibedakan menjadi dua, yaitu orde pertama dan orde kedua. Pada orde pertama memiliki ciri yaitu

terdapat pola yang berulang secara periodik, sedangkan ciri orde kedua pola lokal dan perulangannya tidak terlalu jelas.

3. Ekstraksi fitur berdasarkan warna

Pada ruang warna HSV, nilai *channel hue* yang dikombinasikan dengan nilai *saturation* dan *value* dapat digunakan untuk membedakan antara objek dan warna. Nilai *channel hue* menjelaskan cahaya yang tampak, sedangkan nilai saturation dan value menggambarkan tingkat kecerahan dari suatu warna atau yang disebut dengan ekstraksi fitur berdasarkan warnanya. Untuk mendapatkan ketiga nilai chanel tersebut maka perlu dilakukan konversi citra dari RGB ke HSV terlebih dahulu.

Metode ekstraksi fitur yang sering digunakan yakni, Color Cooccurance Matrix, Canny Edge Detector, Gabor Filters, Discrete Wavelet Transform. Sedangkan tahapan ekstraksi pada penelitian ini yaitu ekstraksi fitur berdasarkan tekstur menggunakan metode ekstraksi GLCM.

2.4. Hue Saturation Value

Selain RGB terdapat representasi nilai warna yakni *Hue*, *Saturation* dan *Value*. *Hue* yang berarti ukuran panjang gelombang pada warna yang dominan berdasarkan persepsi mata manusia. *Saturation* menggambarkan banyaknya cahaya putih yang bercampur pada *channel hue* (Kusumanto, Pambudi, & Tomponu, 2011). Saturasi untuk warna murni yang tidak mengandung cahaya putih bernilai 100 %. Sedangkan saturasi warna murni yang bercampur dengan warna putih bernilai antara 0 hingga 100%. *Brightness* adalah intensitas pemantulan objek yang diterima oleh mata. Intensitas tersebut berarti adanya perubahan warna putih menjadi abu-abu sampai hitam.

Berikut merupakan rumus dan langkah-langkah untuk mengkonversi warna RGB ke HSV (OpenCV, 2017).

Melakukan normalisasi citra dengan membagi nilainya dengan 255 yang ditampilkan pada Persamaan 2.1 sampai 2.3.

$$H = \frac{H}{255} \tag{2.1}$$

$$S = \frac{S}{255} \tag{2.2}$$

$$V = \frac{V}{255} \tag{2.3}$$

Melakukan perhitungan konversi dengan menggunakan Persamaan 2.4 sampai 2.6.

$$V = \max(R, G, B) \tag{2.4}$$

$$S = \begin{cases} \frac{V - \min(R,G,B)}{V} & ,if\ v\ != 0 \\ 0 \quad otherwise \end{cases} \tag{2.5}$$

$$H = \begin{cases} \frac{60(G-B)}{(V-\min(R,G,B))} & if\ V = R \\ \frac{120+60(B-R)}{(V-\min(R,G,B))} & if\ V = G \\ \frac{240+60(R-G)}{(V-\min(R,G,B))} & if\ V = B \end{cases} \quad (2.6)$$

Mengubah citra ke 8 bit image dengan menggunakan Persamaan 2.7 sampai 2.9.

$$V = V\ x\ 255 \quad (2.7)$$

$$S = S\ x\ 255 \quad (2.8)$$

$$H = \frac{H}{2} \quad (2.9)$$

Keterangan:
H: nilai piksel HSV pada *channel* H
S: nilai piksel HSV pada *channel* S
V: nilai piksel HSV pada *channel* V

2.4.1. Fitur Statistik Warna

Perhitungan statistis seperti rata-rata, standar deviasi, skewness dan kurtosis dapat digunakan untuk mendapatkan fitur warna. Penelitian identifikasi tanaman hias telah mengimplementasikannya untuk di terapkan di setiap komponen R, G, dan B. Berikut merupakan rumus-rumus untuk mendapatkan fitur-fitur statistic GLCM pada buku yang ditulis oleh Kadir & Susanto (2013).

1. *Mean*

   Mean merupakan nilai rata-rata, sehingga memberitahu sesuatu tentang kecerahan umum gambar. Citra terang akan memberikan nilai mean yang tinggi, sedangkan citra yang gelap memiliki nilai mean yang rendah.

   $$\mu = \frac{1}{MN}\sum_{i=1}^{M}\ \sum_{j=1}^{N} Pij \quad (2.10)$$

2. Deviasi Standar

   Deviasi standar menjelaskan sesuatu tentang kontras. Deviasi standar mengambarkan penyebaran dalam data. Gambar dengan kontras tinggi maka harus memiliki deviasi standar yang tinggi.

   $$\sigma = \sqrt{\frac{1}{MN}\sum_{i=1}^{M}\ \sum_{j=1}^{N}(Pij - \mu)^{2}} \quad (2.11)$$

3. *Skewness*

   Skewness atau kecondongan menjelaskan ukuran tentang ketidaksimetrisan. Apabila nilai *skewness* negatif menyatakan distribusi miring ke kiri. Sedangkan apabila skewness bernilai positif menyatakan distribusi miring ke kanan.

   $$\theta = \frac{\sum_{i=1}^{M}\ \sum_{j=1}^{N}(Pij-\mu)^{3}}{MN\sigma^{3}} \quad (2.12)$$

4. *Kurtosis*

Kurtosis menunjukkan sebaran data bersifat meruncing atau mengumpul. Apabila nilai *kurtosis* negatif menyatakan distribusi datar. Sedangkan apabila kurtosis bernilai positif menyatakan distribusi bersifat lancip.

$$\gamma = \frac{\sum_{i=1}^{M} \sum_{j=1}^{N} (Pij - \mu)^4}{MN\sigma^4} - 3 \quad (2.13)$$

Keterangan:

M: Panjang citra

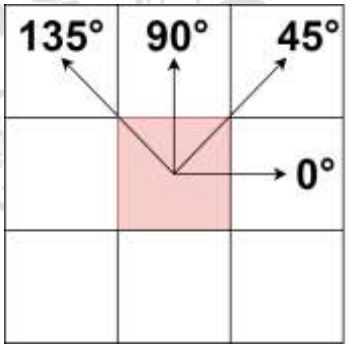
N: Lebar citra

P_{ij}: Nilai piksel pada indeks *i,j*

2.5. Gray Level Co-Occurrence Matrix

GLCM (*Graylevel Cooccurence Matrix*) adalah salah satu metode untuk ekstraksi fitur pada citra digital. GLCM menggunakan matriks untuk menggambarkan frekuensi munculnya 2 pasangan piksel dalam bentuk jarak dan arah dalam suatu citra (Prasetyo, 2011). Metode ini juga dapat digunakan untuk mengklasifisikan dan membedakan objek satu dengan objek yang lain.

GLCM merupakan matriks berukuran GxG dimana dapat dihitung menggunakan Persamaan 14. GLCM menganggap hubungan antara dua piksel pada suatu waktu, piksel referensi dan piksel *offset*. Iterasi pada algoritme melalui setiap piksel referensi secara mendalam dan membandingkan dengan piksel offset lokasi piksel *offset* ditentukan oleh sudut dari piksel referensi (Kadir & Susanto, 2015).



Terdiri dari 135º, 90º, 45º, 0º

Gambar 0.1 Sudut-sudut pada piksel yang digunakan pada GLCM

Sumber : (Madenda, 2015)

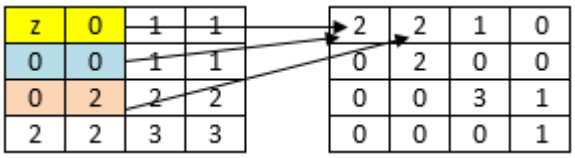
$$G = maks - \min + 1 \quad (2.14)$$

Keterangan:

maks: nilai maksimum dari citra *grayscale*

maks: nilai minimum dari citra *grayscale*

Untuk menemukan nilai GLCM diperlukan penentuan awal matriks yang berbasis dua piksel. Cara yang dilakukan dapat dilihat pada Gambar 2.2.



Gambar 0.2 Menentukan nilai awal *co-occurrence matrix*

Sumber : (Kadir & Susanto, 2013)

Matriks di atas merupakan *matrix framework* yang dapat dijadikan simetrik. Matriks yang simetris didapatkan dengan menambahkan hasil dari transposenya sendiri. Pembentukan matriks *co-occurrence* yang simetris dapat dilihat pada Gambar 2.3.

2	2	1	0
0	2	0	0
0	0	3	1
0	0	0	1

I

\+

2	0	0	0
2	2	0	0
1	0	3	0
0	0	1	1

I'

=

4	2	1	0
2	4	0	0
1	0	6	1
0	0	1	2

I simetriS

Gambar 0.3 Hasil GLCM yang sudah simetris

Sumber : (Kadir & Susanto, 2013)

Agar tidak bergantung pada ukuran citra, nilai GLCM perlu dinormalisasikan sehingga jumlah nilai matriks menjadi 1. Hasil dari normalisasi matriks *co-occurrence* dapat dilihat pada Gambar 2.4.

$\frac{4}{24}$	$\frac{2}{24}$	$\frac{1}{24}$	$\frac{0}{24}$
$\frac{2}{24}$	$\frac{4}{24}$	$\frac{0}{24}$	$\frac{0}{24}$
$\frac{1}{24}$	$\frac{0}{24}$	$\frac{6}{24}$	$\frac{1}{24}$
$\frac{0}{24}$	$\frac{0}{24}$	$\frac{1}{24}$	$\frac{2}{24}$

Gambar 0.4 Hasil normalisasi pada nilai matriks GLCM

Sumber : (Kadir & Susanto, 2013)

2.5.1. Fitur Statistik GLCM

Dibawah ini merupakan fitur-fitur statistic GLCM yang digunakan pada ekstraksi ciri dengan ukuran matriks *co-occurrence* KxK (Madenda, 2015).

1. *Mean*

   Merupakan rata-rata distribusi probabilitas P(i, j) yang dapat dihitung berdasarkan sampel x dan y dimana $\mu_x \neq \mu_y$. Nilai mean dapat dihitung menggunakan Persamaan 2.15 dan 2.16.

$$\mu_x = \sum_{i=0}^{G-1} i \sum_{j=0}^{G-1} P(i,j) \quad (2.15)$$

$$\mu_y = \sum_{j=0}^{G-1} j \sum_{i=0}^{G-1} P(i,j) \quad (2.16)$$

2. *Variance*

Menentukan sebaran nilai atau simpangan terhadap rata-rata data. Apabila varian semakin kecil maka teksturnya semakin homogen dan sebaliknya. Nilai varian dapat dihitung menggunakan Persamaan 2.17 dan 2.18.

$$\sigma_{x^2} = \sum_{i=0}^{G-1}(i - \mu_x)^2 \sum_{j=0}^{G-1} P(i,j) \quad (2.17)$$

$$\sigma_{y^2} = \sum_{j=0}^{G-1}(j - \mu_x)^2 \sum_{i=0}^{G-1} P(i,j) \quad (2.18)$$

3. *Energy*

Energy menunjukkan tingkat keseragaman tekstur. Apabila nilai *energy* semakin tinggi maka homogenitas tekstur semakin besar dan sebaliknya. Nilai *energy* dapat dihitung menggunakan Persamaan 2.19.

$$Energy = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1}(P(i,j))^2 \quad (2.19)$$

4. *Entropy*

Entropy menunjukkan ukuran tingkat keacakan permukaan tekstur akibat gangguan frekuensi. Apabila nilai *entropy* semakin mendekati satu maka tingkat kekasaran tekstur semakin tinggi, dan sebaliknya. Nilai entropy dapat dihitung menggunakan Persamaan 2.20.

$$Entropy = -\sum_{j=0}^{G-1} \sum_{i=0}^{G-1} P(i,j) \log(P(i,j)) \quad (2.20)$$

5. *Contrast*

Contrast menunjukkan nilai intensitas lokal dalam matriks *co-occurrence.* Apabila piksel tetangga memiliki intensitas yang mirip maka *contrast* tekstur sangat rendah. *Contrast* tinggi ditunjukkan dengan variasi intensitas yang tinggi. Sedangkan apabila *contrast* rendah, berarti menunjukkan tekstur yang halus. Nilai *contrast* dapat dihitung menggunakan Persamaan 2.21.

$$Contrast = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1}(i-j)^2 P(i,j) \quad (2.21)$$

6. *Dissimilarity*

Ukuran yang mendefinisikan variasi tingkat intensitas pasangan piksel dalam citra. Jarak yang digunakan adalah menggunakan *city block*. Nilai *dissimilarity* dapat dihitung menggunakan Persamaan 2.22.

$$Dissimilarity = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} |i-j| P(i,j) \quad (2.22)$$

7. *Homogeneity*

Ukuran perulangan struktur tekstur yang bobot nilainya merupakan nilai *invers* dari *contrast*nya. Nilai homogeneity dapat dihitung menggunakan Persamaan 2.23.

$$Homogeneity = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} \frac{P(i,j)}{1+|i-j|} \quad (2.23)$$

8. *Correlation*

Correlation mengukur ketergantungan linier dari tingkat abu-abu dari piksel tetangga pada posisi tertentu. Nilai *correlation* dapat dihitung menggunakan Persamaan 2.24.

$$Correlation = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} \frac{(i-\mu_y)(j-\mu_y)P(i,j)}{\sigma_x \sigma_y} \quad (2.24)$$

Keterangan:

G: Ukuran citra

$P(i,j)$: Nilai piksel pada indeks *i,j*

2.6. Pemilihan Fitur dan Information Gain

Hasil dari ekstraksi fitur dari dataset mungkin berisi terlalu banyak fitur yang akan banyak memakan waktu komputasi. Fitur yang didapatkan tersebut mungkin banyak yang tidak relevan dengan kebutuhan dan adanya fitur yang berulang. Sehingga seleksi fitur diperlukan untuk menghapus fitur seperti itu. Tujuan dari seleksi fitur adalah untuk menemukan satu set fitur minimal yang menghasilkan distribusi kelas sedekat mungkin dengan distriibusi asli yang diperoleh dengan menggunakan semua fitur dan membantu membuat pola lebih mudah dipahami (Azhagusundari & Thanamani, 2013). Beberapa metode pemilihan fitur diantaranya yaitu *Information gain, Chi-Square, Gini Index* dan sebagainya.

Information Gain adalah suatu metode seleksi fitur yang bekerja dengan cara melakukan perangkingan secara sederhana namun telah banyak digunakan dalam berbagai aplikasi salah satunya yaitu pada analisis data citra (Chormunge & Jena, 2016). Dalam penelitian Shaltout, et al. (2014) menyebutkan bahwa *Information Gain* mendeteksi fitur-fitur yang paling relevan berdasarkan kelas-kelas tertentu dengan melakukan perhitungan nilai *entropy*. *Entropy* adalah ukuran dari ketidakpastian kelas menggunakan probabilitas dari suatu atribut tertentu.

Langkah-langkah mengubah data tunggal menjadi data berkelompok menggunakan distribusi frekuensi menurut Darwiyo, Binawan, & Junaedi (2017) dapat dihitung denggan menggunakan Persamaan 2.25 sampai 2.27.

1. Menghitung jangkauan (J)

$$J = nilai\ maksimum - nilai\ minimum \quad (2.25)$$

2. Mencari banyak kelas (K)

$$K = 1 + 3{,}3\ x \log n \quad (2.26)$$

dimana n adalah jumlah data

3. Menghitung panjang kelas (C)

$$C = \frac{J}{K} \quad (2.27)$$

4. Memasukkan data ke kategori

Tahapan dalam proses perhitungan *Information Gain* dapat dihitung menggunakan persamaan 2.28 sampai 2.29. (Firmahsyah & Gantini, 2016).

a. Cari nilai entropi seluruh kelas dengan Persamaan 2.29.

$$Entropi(S) = -\sum_i^c Pi \log 2\, Pi \quad (2.28)$$

c: jumlah kelas

Pi: jumlah sampel setiap kelas i

b. Cari nilai gain setiap fitur A dengan rumus Persamaan 2.30.

$$Gain(S, A) = Entropy(S) - \sum_{c€values(A)} \frac{|Sv|}{|S|} Entropy(Sv) \quad (2.29)$$

Keterangan

A: fitur

v: nilai untuk fitur A

Values(A) : nilai-nilai untuk fitur A

|Sv |: jumlah sampel v

| S |: jumlah semua data

Entropy(Sv): *entropy* untuk data yang memiliki nilai v

2.7. Klasifikasi

Klasifikasi mempunyai fungsi untuk memasukkan data ke dalam kelas tertentu. Dalam klasifikasi ada hal yang harus dilakukan, yaitu pembangunan model *prototipe* untuk dijadikan memori untuk penyimpanan data dan melakukan klasifikasi untuk mengetahui letak dari objek data tersebut disimpan atau dikelaskan (Prasetyo, 2012). Contoh metode klasifikasi antara lain, *K-nearest Neighbors*, *Support Vector Machines* dan *Naive Bayes*.

2.7.1. Algoritme K–Nearest Neighbor

K-Nearest Neighbor (KNN) merupakan salah satu metode klasifikasi yang menggunakan jarak terdekat sebagai pendekatannya. Apabila terdapat *query* baru maka akan diklasifikasikan berdasarkan kedekatan jarak yang paling banyak dari tiap kategorinya. Teknik untuk mencari tetangga terdekat yang dapat dilakukan dengan menggunakan jarak *Euclidean*.

Jarak Euclidean adalah cara mencari jarak antara dua titik dalam ruang dua dimensi. Persamaan jarak Euclidean didefinisikan pada Persamaan 2.30. Hastuti, Dewi, & Widodo (2017).

$$\boldsymbol{d_i} = \sqrt{\sum_{i=1}^{n} (\boldsymbol{p_i} - \boldsymbol{q_i})^2} \quad (2.30)$$

Keterangan :

$\boldsymbol{d_i}$: hasil perhitungan jarak euclidean

$\boldsymbol{p_i}$: data latih

$\boldsymbol{q_i}$: data uji

n: jumlah data

i: indeks antara 0 sampai dengan n-1

Sebelum melakukan perhitungan jarak, dilakukan normalisasi data terlebih dahulu untuk menyamakan rentang data antar fitur. Normalisasi data menurut (Patro & Sahu, 2015) didefinisikan dengan Persamaan 2.31.

$$A' = \left(\frac{A - \min value\ of\ A}{\max\ value\ of\ A - \min value\ of\ A}\right) * (D - C) + C$$

(2.31)

Keterangan:

A': nilai hasil normalisasi

A: nilai asli

C, D: batas yang telah ditentukan sebelumnya

2.8. Akurasi

Akurasi adalah nilai yang digunakan untuk menunjukkan berapa besar ketepatan tingkat pengujian dengan nilai yang telah diprediksikan. Selain itu, nilai akurasi menunjukkan keberhasilan sistem yang telah dibangun. Akurasi disini menunjukan tingkat keakuratan metode yang digunakan dengan cara membandingkan antara hasil pengujian dengan data asli. Rumus untuk melakukan perhitungan akurasi ditunjukkan pada Persamaan 2.32.

$$akurasi = \frac{jumlah\ klasifikasi\ benar}{jumlah\ data\ uji}\ x\ 100\ \% \qquad (2.32)$$

Jumlah klasifikasi benar yaitu ketika data uji dan kelas sebenarnya menghasilkan hasil prediksi yang benar oleh metode klasifikasi kemudian dihitung jumlah hasil prediksi yang benar tersebut. Sedangkan jumlah data uji yaitu jumlah seluruh data yang akan diprediksi.

BAB 3 METODOLOGI

Bab ini berisi metode-metode dan tahapan yang digunakan dalam menyelesaikan penelitian seleksi fitur *Information Gain* untuk klasifikasi citra makanan berdasarkan HSV dan GLCM.

3.1 Tipe Penelitian

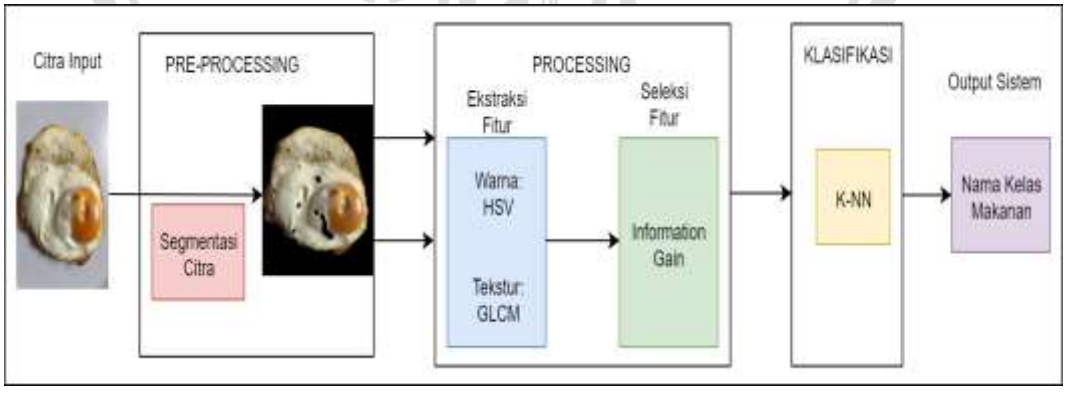
Penelitian ini menggunakan tipe penelitian non implementatif analitik, yaitu penelitian yang menitikberatkan pada kegiatan analisis dengan menjelaskan hubungan antar elemen pada objek penelitian.

3.2 Strategi Penelitian

Strategi penetilian pada penelitian ini menguraikan mengenai alur berjalannya program yang ditunjukkan pada Gambar 3.1. Masukan pada program merupakan citra makanan tunggal. Selanjutnya citra makanan tunggal tersebut akan memasuki tahap *pre-processing* untuk mendapatkan citra yang kualitasnya lebih baik dari semula.

Setelah melalui tahap *pre-processing*, tahap selanjutnya yaitu mengolah citra hasil *pre-processing* tadi ke tahap *processing*. Tahap *processing* terbagi menjadi dua tahapan, yaitu ekstraksi fitur dan seleksi fitur. Ekstraksi fitur digunakan untuk mendapatkan fitur-fitur statistik menggunakan ekstraksi warna HSV dan tekstur GLCM, kemudian dilakukan seleksi fitur *Information Gain* untuk menyeleksi fitur-fitur yang dihasilkan pada proses ekstraksi fitur agar mendapatkan fitur-fitur yang relevan. Setelah mendapatkan fitur-fitur yang relevan kemudian masuk ke tahap berikutnya, yaitu klasifikasi.

Klasifikasi yang digunakan pada penelitian ini menggunakan metode KNN. Hasil keluaran dari program akan mendeteksi jenis makanan sesuai pada citra yang menjadi input dengan memberikan label pada masing-masing citra dengan nama jenis makanan.



Gambar 0.1 Strategi penelitian

3.3 Lokasi Penelitian

Penelitian ini dilakukan di Ruang Grup Riset Computer Vision di Ruang F9.3 Gedung F lantai 9, Fakultas Ilmu Komputer, Universitas Brawijaya.

3.4 Metode Pengumpulan Data

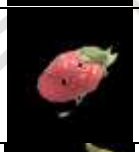
Tahap pengumpulan data memaparkan tentang data yang digunakan dalam penelitian ini. Data dikumpulkan secara primer yaitu dengan melakukan pengambilan data secara manual menggunakan *handphone*. Pada penelitian ini objek yang digunakan yaitu:

1. Penelitian ini menggunakan objek berupa citra makanan tunggal yang berbentuk padat berjumlah dua puluh satu jenis makanan. Dimana dua puluh satu jenis makanan diberi kode dan label seperti yang dituliskan pada tabel 3.1.
2. Cahaya matahari yang ada pada gedung ruang Grup Riset *Computer Vision* di gedung F lantai 9, Fakultas Ilmu Komputer, Universitas Brawijaya berperan sebagai pencahayaan dalam pengambilan citra makanan.
3. Pengambilan citra dilakukan menggunakan *handphone Iphone* 7+ dengan kamera utama 12 megapiksel, *thelephoto* bukaan f/1.8, dan *wide-angle* bukaan f/2.8.
4. Objek makanan yang diambil gambarnya, dilakukan dengan beberapa teknik seperti diambil dengan tegak lurus dengan ketinggian yang berbeda dan kemiringan yang berbeda.
5. Objek makanan diambil dengan bagian yang berbeda-beda meliputi 1 bagian, ½ bagian, ¼ bagian dan ¾ bagian.

Tabel 0.1 Kode dan label kelas

Kode Kelas	Nama Label Kelas	Contoh Data	Contoh Data Tesegmentasi
0	Donat		
1	Roti Gandum		
2	Roti Tawar		
3	Telor		

Tabel 0.2 Kode dan label kelas (lanjutan)

Kode Kelas	Nama Label Kelas	Contoh Data	Contoh Data Tesegmentasi
5	Rendang		
6	Timun		
7	Selada		
8	Tomat		
9	Strawberry		
10	Pisang Hijau		
11	Pisang Kuning		
12	Jeruk		
13	Nasi Merah		
14	Oreo		
15	Beng-beng		
16	Soba Mie		

Tabel 0.3 Kode dan label kelas (lanjutan)

17	Gerry Salut		
18	Biskuat		
19	Milo Nuggets		
20	Genji Pie		

Tabel 0.4 Contoh data training dan testing

Kode Kelas	Data Training	Data Testing
0		
3		
7		
18		

3.5 Teknik Analisis Data

Setelah melakukan pengumpulan data selanjutnya yaitu melakukan analisis data. Data yang telah dikumpulkan dilakukan analisis untuk mengetahui kinerja dari sistem. Analisis dilakukan dengen membandingkan terhadap kelas sebenarnya guna mengetahui akurasi. Nilai *k* dilakukan untuk melakukan analisis berdasarkan kombinasi fitur dan *Information Gain*.

3.6 Peralatan Prendukung

Pada pengembangan sistem diperlukan peralatan yang mendukung penelitian. Peralatan yang mendukung penelitian ini meliputi:

1. Spesifikasi *hardware*

   Laptop dengan *Processor* Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.71 GHz, Memori(RAM) 4,00 GB, *Hardisk* 1 TB, Ukuran layar 14 *inch*.

2. Spesifikasi *software*

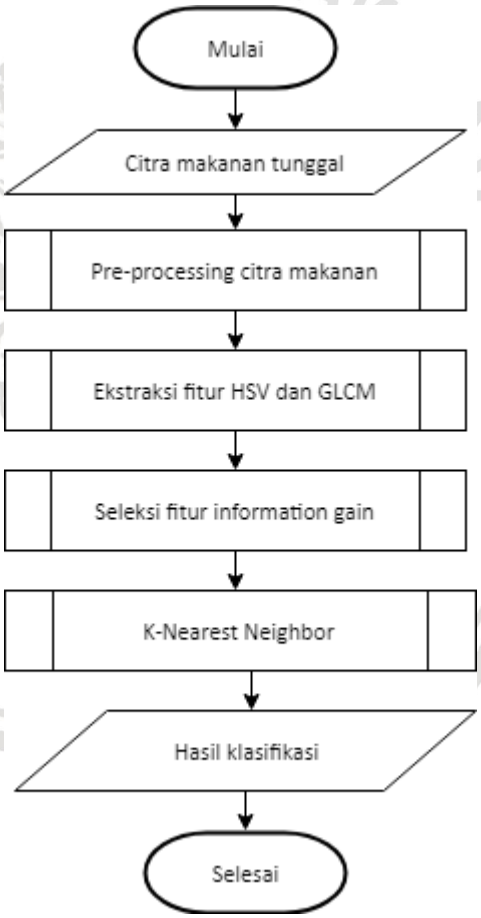
   - *Microsoft Windows* 10
   - *Python* 3.6

BAB 4 PERANCANGAN

Pada bab ini dijelaskan tahapan dalam perancangan sistem seleksi fitur *Information Gain* berdasarkan HSV dan GLCM pada citra makanan hingga menampilkan hasil klasifikasinya.

4.1 Perancangan Algoritme dan Proses Implementasi

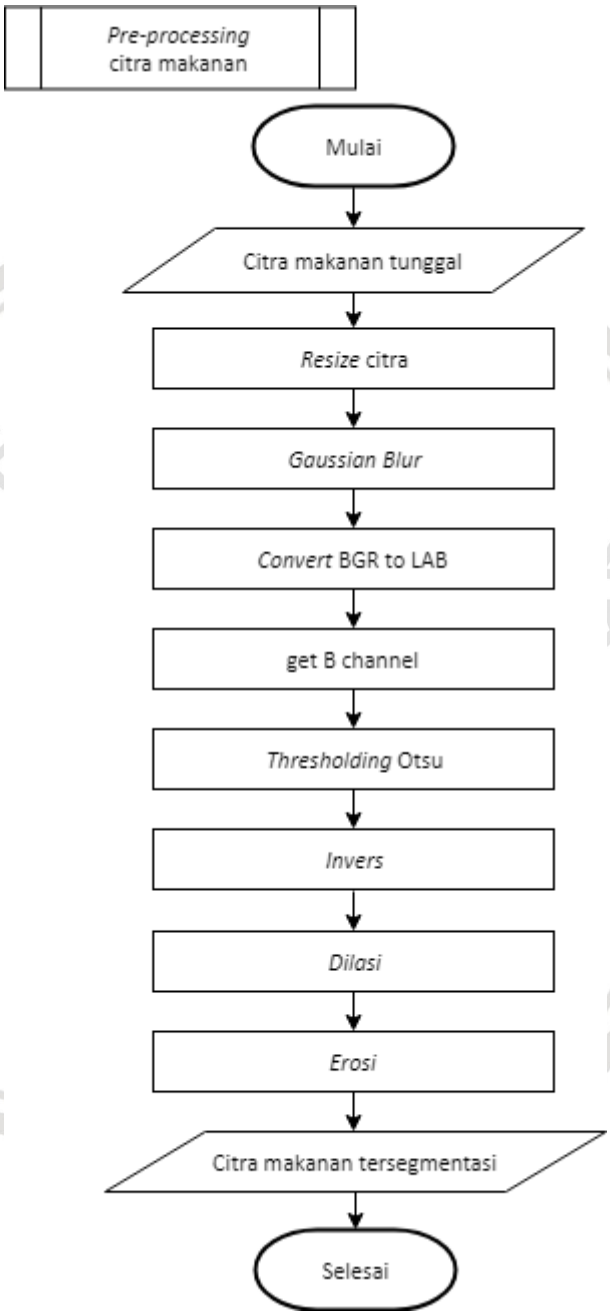
Terdapat empat proses utama yang diimplementasikan pada sistem di penelitian ini, yaitu *pre-processing*, ekstraksi fitur (warna dan tekstur), seleksi fitur dan klasifikasi KNN. Pada tahap *pre-processing*, citra masukan akan diproses sehingga menghasilkan keluaran citra yang digunakan untuk ekstraksi fitur (warna dan tekstur), kemudian fitur-fitur yang telah didapatkan akan diseleksi untuk mendapatkan fitur-fitur yang relevan dengan menggunakan seleksi fitur *Information Gain*. Setelah mendapatkan fitur-fitur yang relevan selanjutnya yaitu melakukan klasifikasi menggunakan KNN menggunakan fitur-fitur tersebut. Alur proses implementasi ditunjukkan pada Gambar 4.1.



Gambar 0.1 Alur perancangan sistem

4.1.1 Perancangan Pre-Processing

Pre-processing merupakan proses yang paling awal pada pengolahan citra yang bertujuan untuk memperbaiki kualitas citra sebelum masuk ke tahap selanjutnya. Pada penelitian ini *pre-processing* digunakan pada citra data latih dan data uji dimana keduanya memiliki alur yang sama. Alur dari *pre-processing* ditunjukkan pada Gambar 4.2.

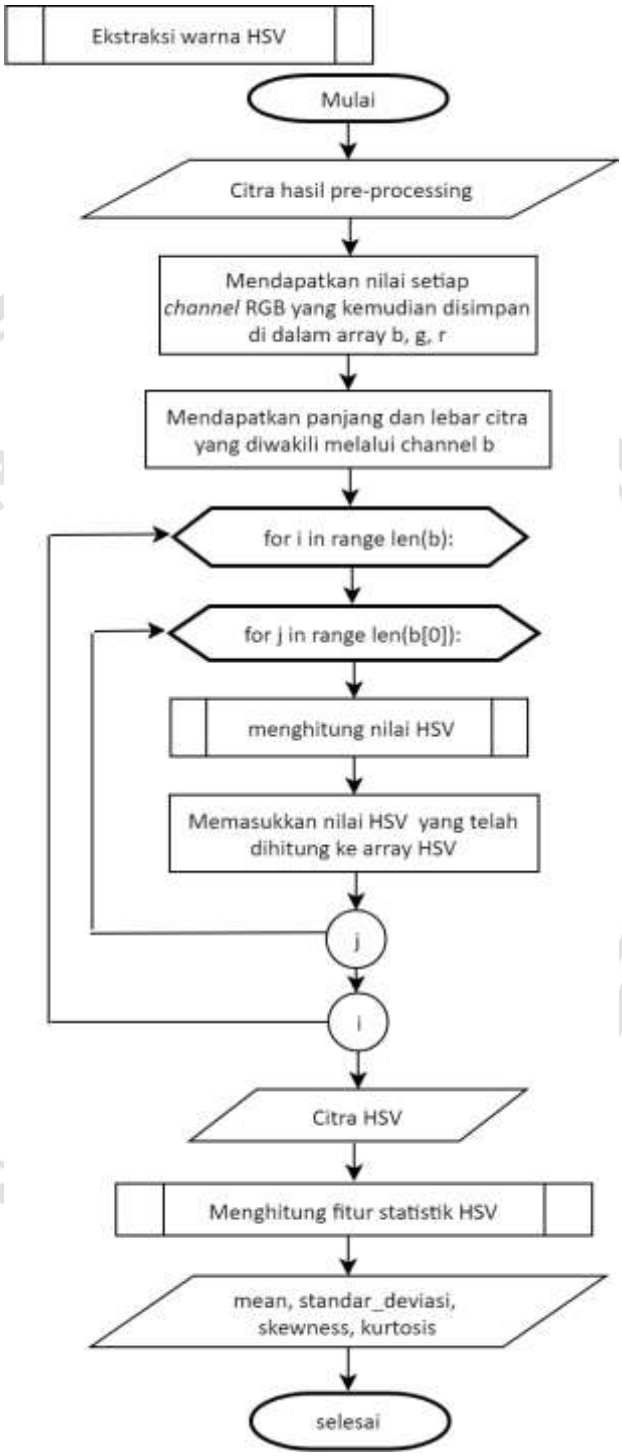


Gambar 0.2 Diagram alir *pre-processing*

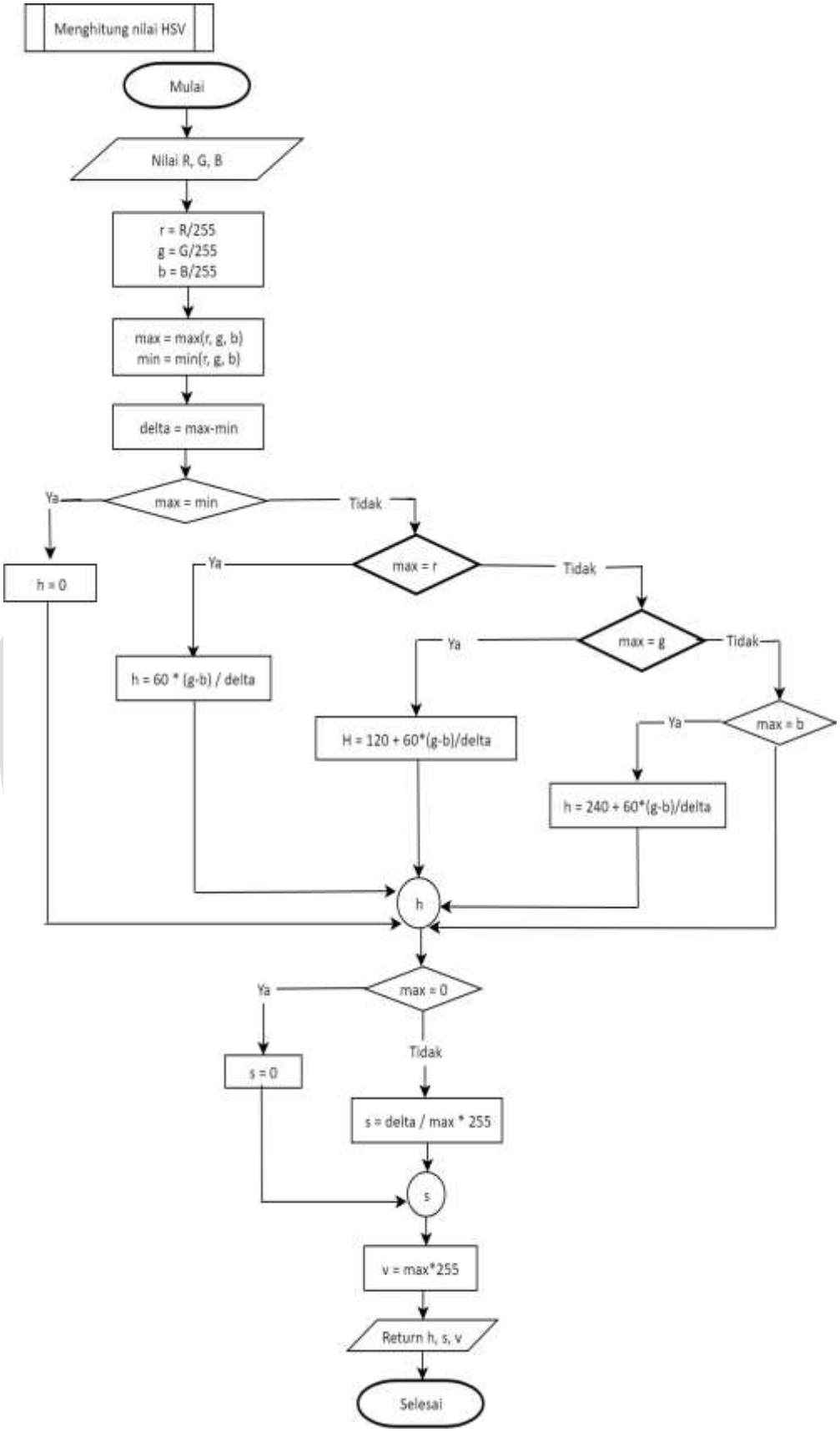
4.1.2 Perancangan Ekstraksi Warna HSV

Ekstraksi fitur warna merupakan proses mendapatkan fitur-fitur statistik bersasarkan *Color Moment* HSV, diagram alirnya ditunjukan pada Gambar 4.3.

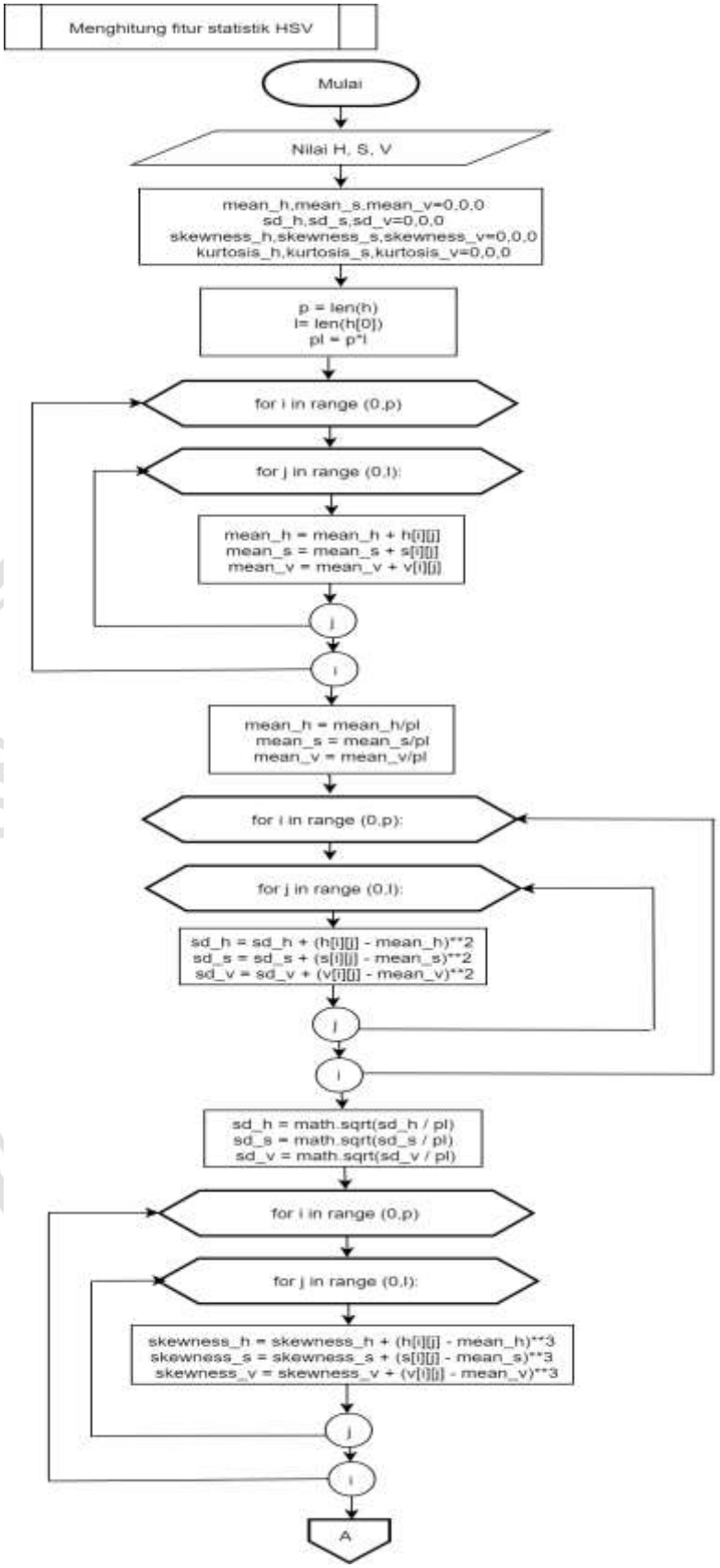
Mendapatkan citra HSV terlebih dahulu dengan melakukan konversi warna RGB ke HSV seperti yang ditunjukkan pada Gambar 4.4 menggunakan citra makanan yang telah dilakukan *pre-processing*. Kemudian dilanjutkan dengan melakukan perhitungan fitur-fiturnya dari citra yang telah dilakukan konversi seperti ditunjukkan pada Gambar 4.5 dan Gambar 4.6.



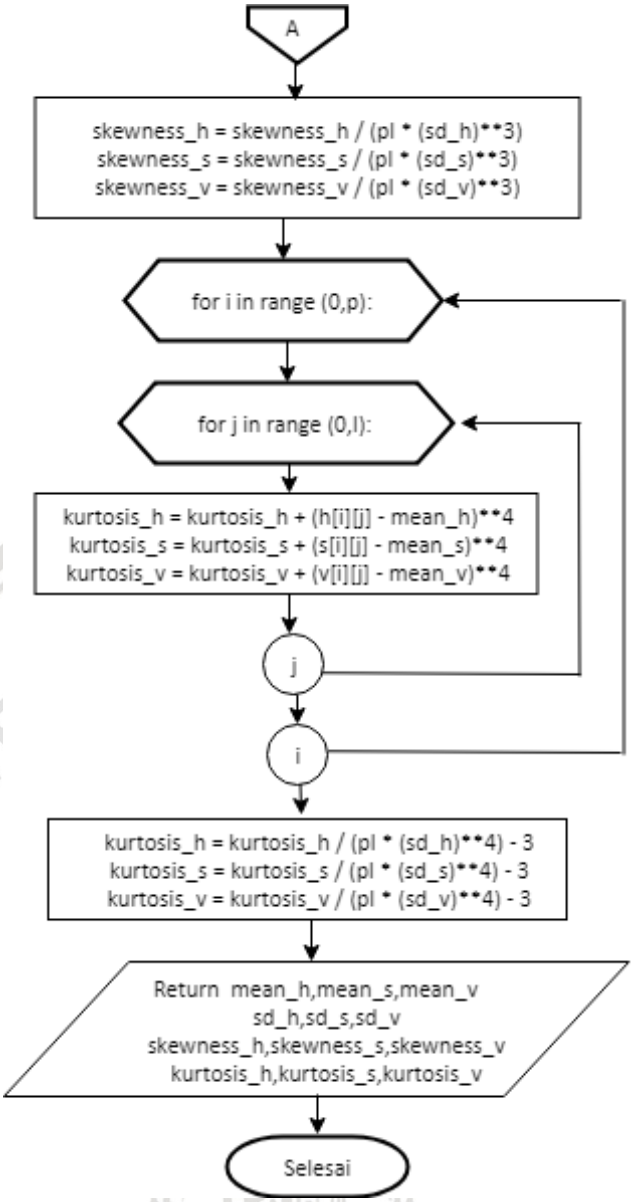
Gambar 0.3 Diagram alir ekstraksi warna HSV



Gambar 0.4 Konversi RGB ke HSV



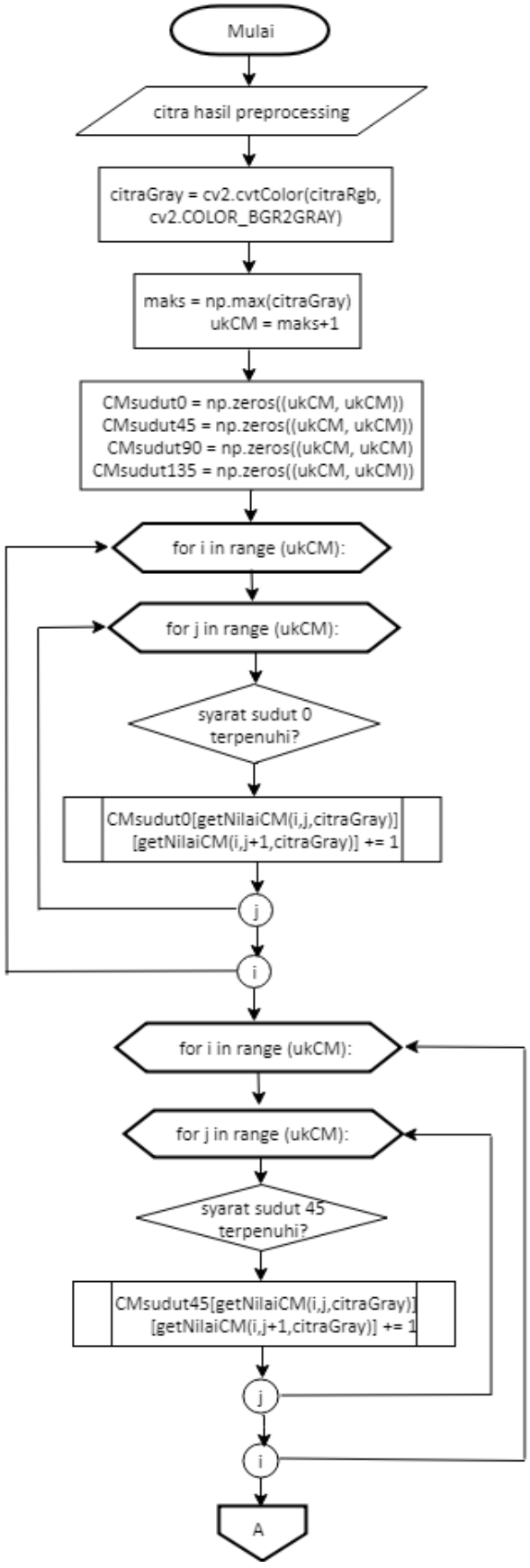
Gambar 0.5 Mendapatkan fitur statistik HSV



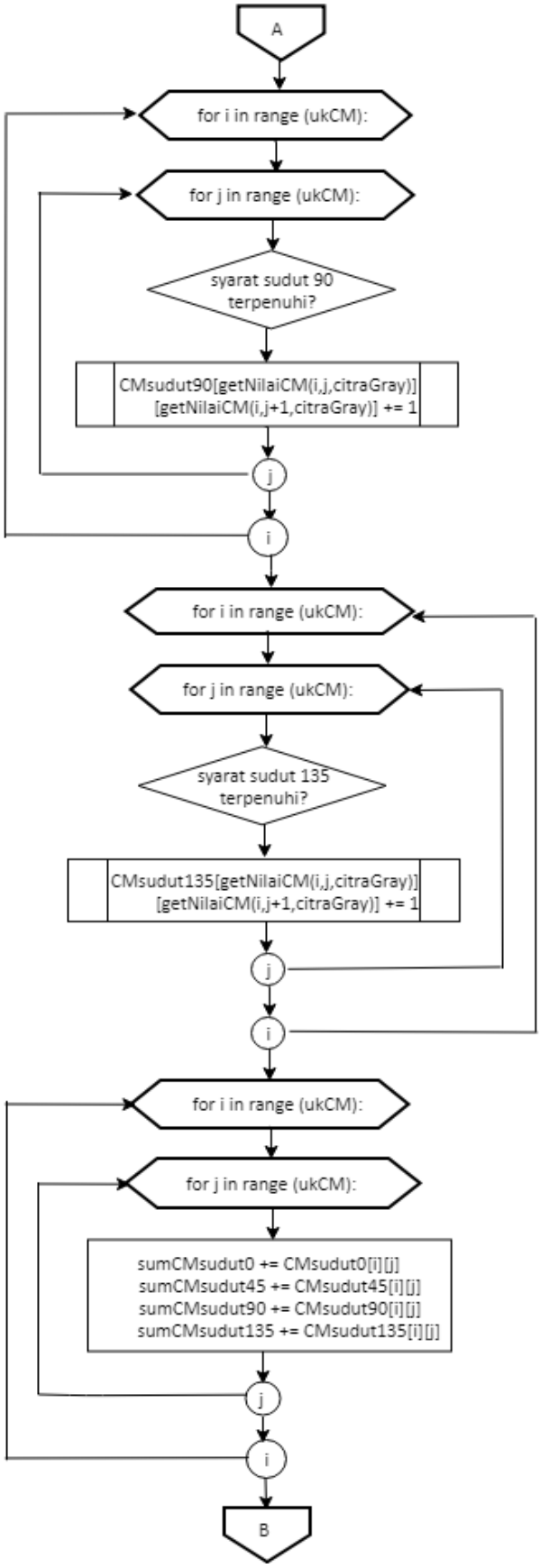
Gambar 0.6 Mendapatkan fitur statistik HSV (lanjutan)

4.1.3 Perancangan Ekstraksi Tekstur GLCM

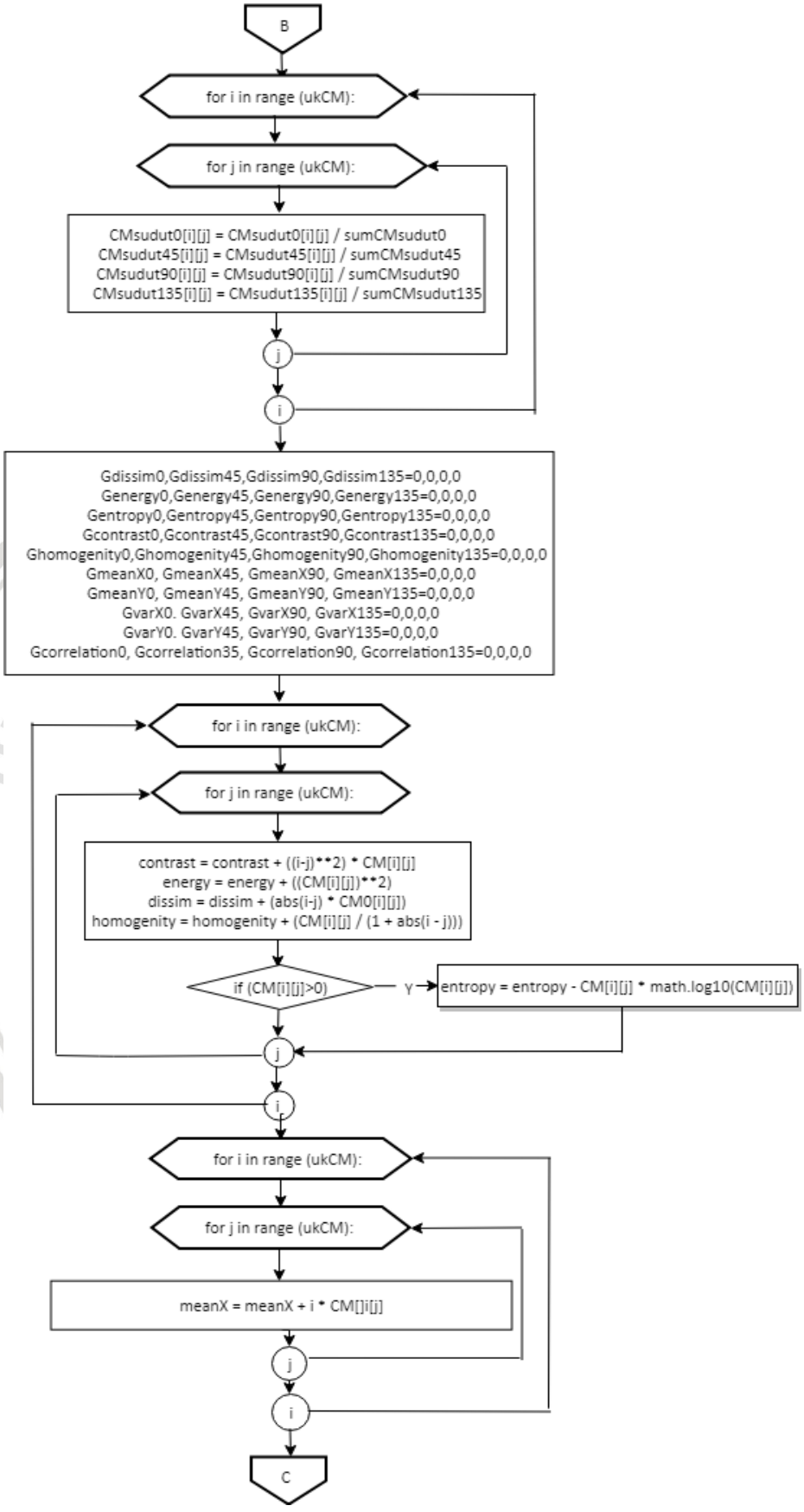
Proses ekstraksi tekstur GLCM diawali dengan mendapatkan matriks *co - ocurence*nya terlebih dahulu, kemudian dilakukan normalisasi, hasil dari normalisasi matriks *co – occurrence* kemudian digunakan untuk mendapatkan fitur-fitur statistiknya. Pada penelitian ini terdapat sepuluh fitur yang di ekstraksi dengan menggunakan 4 sudut yaitu 0°, 45°, 90°, dan 135°, sehingga keseluruhannya terdapat 40 fitur yang di ekstraksi. Diagram alir ekstraksi tekstur GLCM dan proses untuk mendapatkan matriks CM sesuai dengan ukurannya ditunjukkan pada Gambar 4.7 sampai dengan Gambar 4.11.



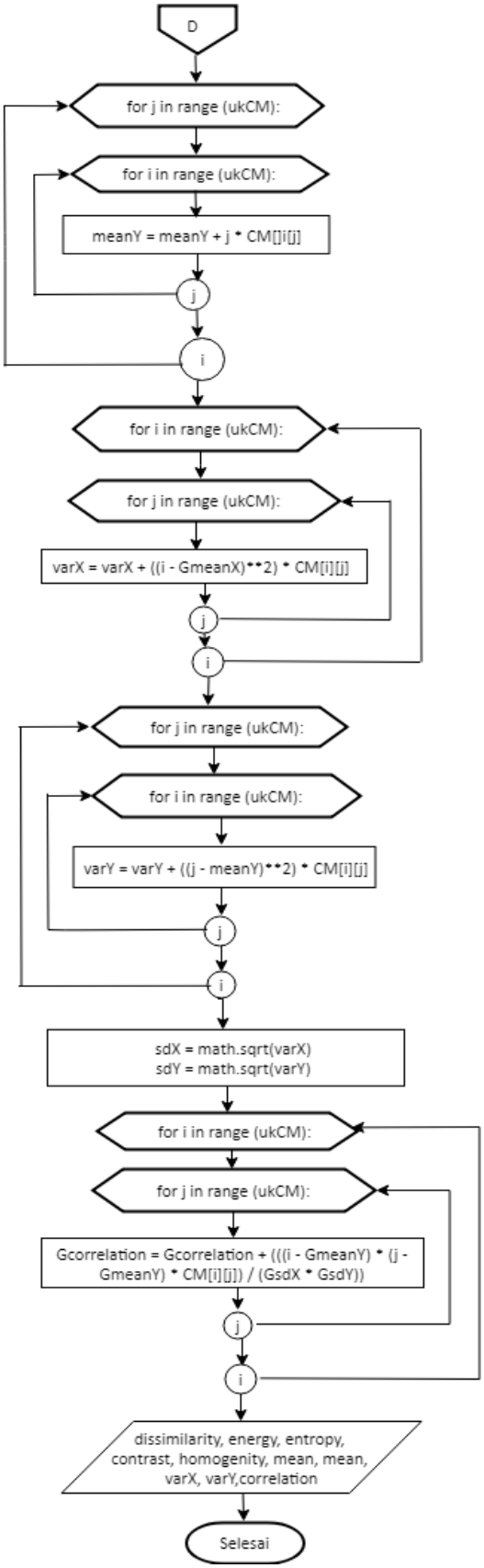
Gambar 0.7 Ekstraksi tekstur GLCM



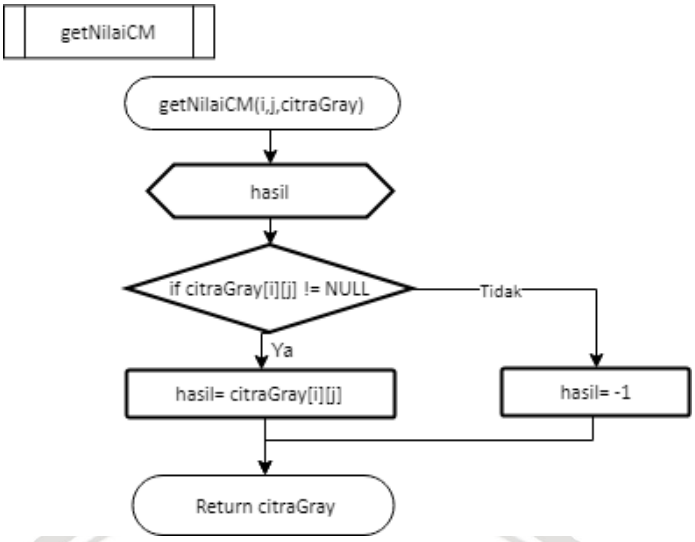
Gambar 0.8 Ekstraksi tekstur GLCM (lanjutan)



Gambar 0.9 Ekstraksi tekstur GLCM (lanjutan)



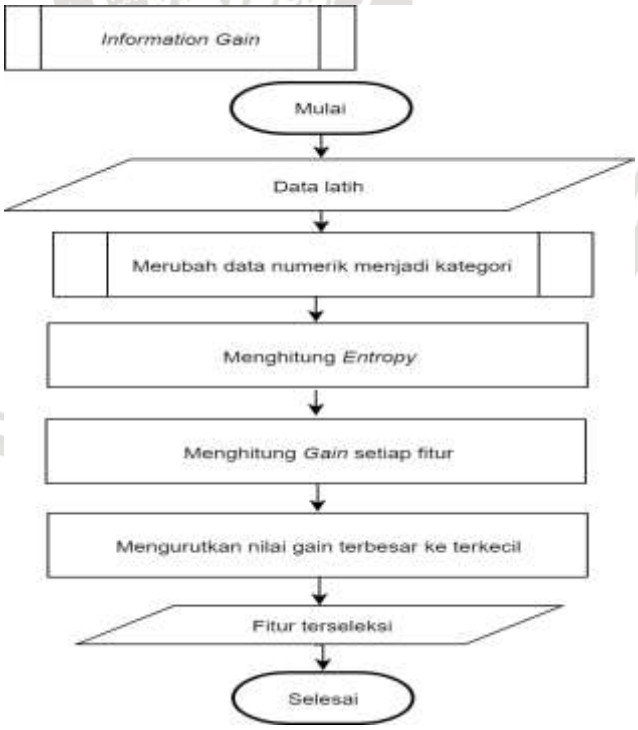
Gambar 0.10 Ekstraksi tekstur GLCM (lanjutan)



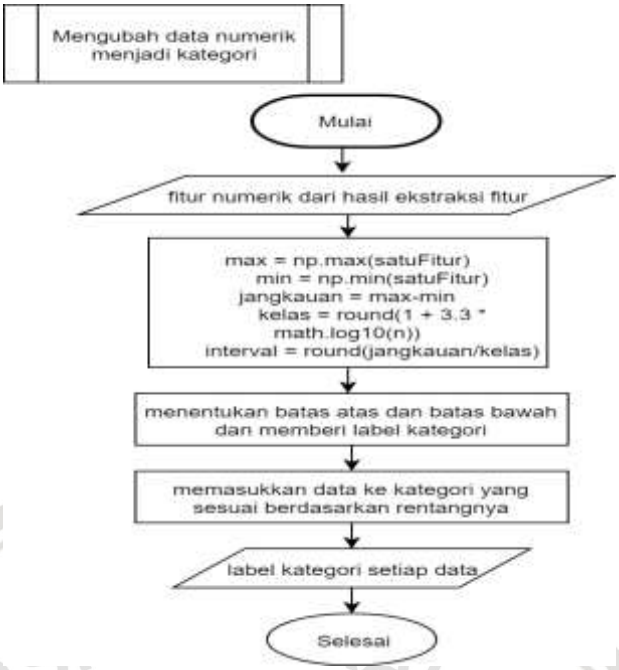
Gambar 0.11 Diagram alir mendapatkan matriks CM sesuai ukurannya

4.1.4 Perancangan Information Gain

Proses seleksi fitur ditentukan oleh besarnya nilai *gain* pada setiap fitur. Nilai gain diurutkan dari yang terbesar kemudian diambil sejumlah atribut yang diinginkan. Pada penelitian ini, sebelum dilakukan perhitungan untuk menghitung nilai *entropy* dan *gain*nya, terlebih dulu melakukan mengubahan data numerik ke kategori karena hasil dari ekstraksi fitur berupa data-data numerik. Untuk melakukan perubahan data prosesnya dijelaskan pada Gambar 4.13 sedangkan keseluhan tahapnya dijelaskan pada Gambar 4.12.



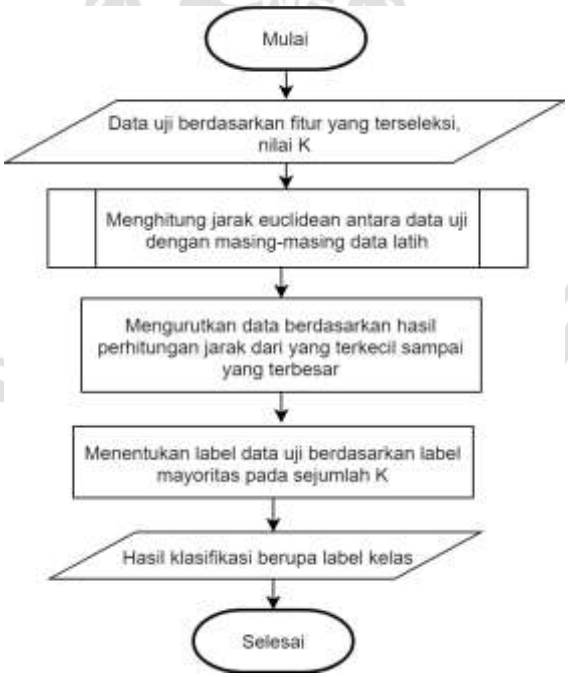
Gambar 0.12 Diagram alir *Information Gain*



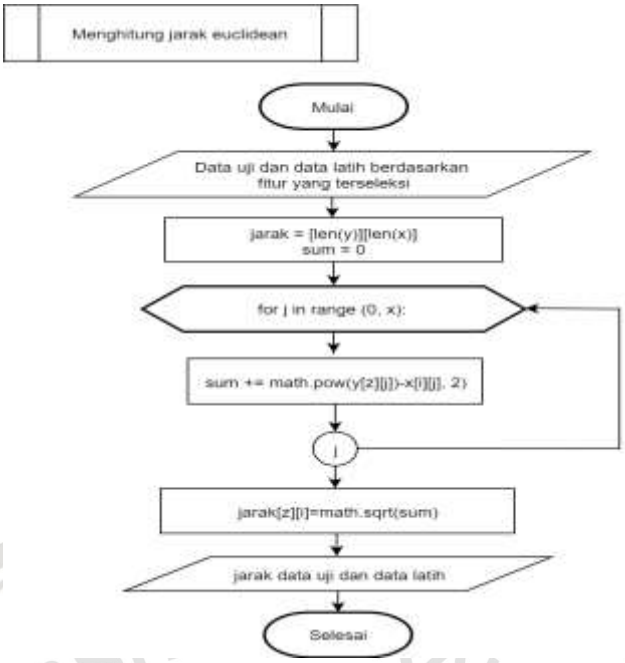
Gambar 0.13 Diagram alir pengubahan data ke kategori

4.1.5 Perancangan K – Nearest Neighbor

KNN merupakan metode klasifikasi yang dilakukan dengan memilih jarak terdekat dari kelas-kelas yang ada untuk menentukan kelas dari data baru. Pada penelitian ini, metode yang digunakan untuk mengukur jarak adalah *Euclidean Distance* yang alurnya dapat dilihat pada Gambar 4.15. Alur klasifikasi KNN ditunjukkan pada Gambar 4.14.



Gambar 0.14 Diagram alir klasifikasi KNN



Gambar 0.15 Diagram alir menghitung jarak Euclidean

4.2 Manualisasi

Tahapan manualisasi memberikan gambaran mengenai proses komputasi tanpa menggunakan sistem. Langkah-langkah perhitungannya dilakukan secara detail dengan perhitungan menggunakan Ms. *Excel* dan *calculator*. Perhitungan manual menggunakan citra telur berukuran 10x10 untuk *pre-processing* sedangkan untuk ekstraksi fitur menggunakan citra telur berukuran 3x3. Gambar 4.16 menunjukkan citra telur yang digunakan untuk perhitungan manual.

Gambar 0.16 Citra perhitungan manual

4.2.1 Manualisasi Pre-Processing

Tahap pre-processing merupakan tahap pertama yang meliputi *filtering*, *konversi RGB ke LAB*, *thresholding* dan morfologi *closing*. Hasil dari tahap ini merupakan citra tersegmentasi yang akan digunakan untuk proses selanjutnya. Pada Tabel 4.1 ditunjukkan nilai piksel dari citra awal yang digunakan.

Tabel 0.1 Nilai B/G/R pada citra manualisasi

168/ 152/ 142	177/ 160/ 150	176/ 158/ 147	177/ 159/ 159	183/ 163/ 156	193/ 173/ 156	196/ 177/ 160	197/ 180/ 164	192/ 176/ 161	136/ 125/ 118
177/ 161/ 149	176/ 160/ 147	175/ 158/ 145	150/ 156/ 155	131/ 151/ 159	146/ 150/ 151	186/ 165/ 150	187/ 167/ 151	185/ 168/ 153	168/ 151/ 140
178/ 161/ 150	176/ 159/ 146	158/ 161/ 157	133/ 158/ 162	126/ 151/ 158	132/ 161/ 173	132/ 157/ 171	176/ 159/ 147	179/ 162/ 147	175/ 157/ 144
177/ 160/ 146	173/ 157/ 142	141/ 160/ 160	116/ 141/ 149	152/ 169/ 170	67/ 136/ 184	43/ 116/ 182	114/ 145/ 166	175/ 157/ 143	175/ 156/ 141
178/ 160/ 144	158/ 147/ 136	117/ 140/ 146	159/ 174/ 172	152/ 165/ 167	72/ 122/ 170	25/ 91/ 168	86/ 131/ 166	173/ 154/ 139	175/ 155/ 139
179/ 160/ 145	163/ 156/ 132	123/ 141/ 144	161/ 175/ 171	140/ 158/ 160	123/ 140/ 154	52/ 89/ 146	111/ 129/ 142	172/ 151/ 136	176/ 154/ 138
178/ 161/ 145	174/ 156/ 140	122/ 128/ 125	148/ 160/ 156	160/ 174/ 170	130/ 146/ 147	152/ 161/ 157	106/ 119/ 129	171/ 151/ 134	175/ 154/ 138
179/ 161/ 148	174/ 157/ 142	135/ 126/ 123	93/ 110/ 120	137/ 148/ 143	136/ 146/ 142	109/ 116/ 117	151/ 133/ 120	173/ 153/ 137	176/ 156/ 141
181/ 164/ 152	174/ 157/ 144	169/ 151/ 137	92/ 93/ 102	78/ 78/ 86	98/ 93/ 90	156/ 137/ 122	172/ 152/ 136	173/ 155/ 142	176/ 158/ 144
194/ 177/ 165	173/ 157/ 144	170/ 153/ 138	167/ 150/ 134	169/ 148/ 131	169/ 149/ 131	169/ 150/ 133	169/ 152/ 147	171/ 154/ 142	170/ 153/ 141

Metode yang digunakan pada penelitian ini untuk melakukan *filtering* yaitu *Gaussian* blur *filter* dengan ukuran mask 3x3. Hasil *filtering* dari proses ini ditunjukkan pada Tabel 4.2.

Tabel 0.2 Nilai piksel perhitungan *Gaussian* filter B/G/R

174/ 158/ 142	175/ 158/ 147	173/ 158/ 148	165/ 158/ 151	162/ 158/ 153	172/ 163/ 154	186/ 169/ 155	191/ 172/ 157	180/ 164/ 150	170/ 155/ 143
176/ 159/ 148	175/ 159/ 148	168/ 159/ 150	155/ 157/ 154	148/ 156/ 156	157/ 159/ 157	173/ 165/ 157	183/ 158/ 154	180/ 163/ 149	174/ 157/ 145
176/ 160/ 147	172/ 160/ 148	156/ 158/ 153	139/ 156/ 158	130/ 154/ 162	124/ 152/ 168	132/ 152/ 165	157/ 156/ 155	174/ 159/ 148	176/ 159/ 145

Tabel 4.2 Nilai piksel perhitungan *Gaussian* filter B/G/R (lanjutan)

174/ 158/ 144	164/ 165/ 146	145/ 155/ 153	137/ 156/ 160	127/ 155/ 167	94/ 140/ 174	82/ 131/ 173	120/ 142/ 160	162/ 154/ 147	175/ 157/ 143
170/ 155/ 141	157/ 151/ 142	140/ 151/ 150	143/ 160/ 162	133/ 156/ 167	88/ 130/ 168	64/ 114/ 165	102/ 130/ 156	155/ 149/ 144	174/ 155/ 139
172/ 154/ 140	157/ 149/ 139	141/ 149/ 146	146/ 161/ 160	141/ 159/ 163	111/ 136/ 158	89/ 120/ 152	112/ 128/ 145	156/ 146/ 139	174/ 153/ 137
157/ 157/ 142	162/ 150/ 138	140/ 142/ 137	137/ 149/ 148	141/ 156/ 155	131/ 146/ 150	120/ 133/ 142	131/ 133/ 135	159/ 144/ 135	174/ 152/ 137
177/ 159/ 145	167/ 152/ 140	140/ 135/ 130	120/ 127/ 128	122/ 131/ 131	127/ 134/ 131	133/ 133/ 128	148/ 133/ 130	166/ 149/ 136	174/ 155/ 139
179/ 162/ 148	173/ 156/ 144	152/ 140/ 131	125/ 120/ 119	117/ 114/ 115	128/ 122/ 115	147/ 134/ 123	163/ 134/ 132	171/ 153/ 139	174/ 155/ 142
180/ 164/ 151	176/ 159/ 146	160/ 146/ 134	138/ 127/ 120	128/ 117/ 114	138/ 125/ 114	157/ 140/ 126	169/ 140/ 137	172/ 154/ 141	172/ 155/ 142

Tahap selanjutnya adalah *thresholding* dimana *thresholding* memisahkan objek dan latar belakang berdasarkan gelap terang dalam citra. Sebelum dilakukan *thresholding* citra rgb terlebih dulu dilakukan konversi ke ruang warna L*a*b untuk mendapatkan channel B. Matriks *channel* B yang didapatkan dapat dilihat pada Tabel 4.3.

Tabel 0.3 Nilai piksel pada channel B dari citra L*a*b

118	118	119	123	125	122	118	117	118	119
118	118	122	129	132	129	123	119	118	118
118	120	128	137	142	145	140	127	119	118
118	123	133	138	144	157	161	142	123	117
118	124	134	137	142	156	163	147	124	116
117	123	132	136	138	145	150	139	122	115
117	120	128	134	136	137	136	129	120	115
117	119	125	132	133	131	127	122	117	116
118	118	120	125	126	124	120	117	117	117
118	118	119	121	121	1201	117	116	117	118

Berdasarkan nilai piksel channel B, dilakukan *thresholding* menggunakan metode *ostu*. Hasil yang didapatkan dari proses *thresholding* dapat dilihat pada Tabel 4.4.

Tabel 0.4 Nilai piksel *Thresholding Otsu*

255	255	255	255	255	255	255	255	255	255
255	255	255	255	0	255	255	255	255	255
255	255	255	0	0	0	0	255	255	255
255	255	0	0	0	0	0	0	255	255
255	255	0	0	0	0	0	0	255	255
255	255	0	0	0	0	0	0	255	255
255	255	255	0	0	0	0	255	255	255
255	255	255	0	0	0	255	255	255	255
255	255	255	255	255	255	255	255	255	255
255	255	255	255	0	255	255	255	255	255

Tahap selajutnya yaitu mendapatkan citra tersegmentasi yang dimulai dengan melakukan *invers* dari hasil *thresholding*. *Invers* dilakukan karena hasil dari *thresholding* merupakan kebalikan dari yang diinginkan. *Invers* yaitu mengubah nilai piksel yang bernilai 0 menjadi 255 dan sebaliknya, sehingga dengan proses ini dihasilkan matriks seperti Tabel 4.5. Kemudian dilakukan operasi *morfologi closing* untuk menyempurnakan hasil dari *thresholding*. Hasil yang di dapatkan dari operasi *morfologi closing* dapat dilihat di Tabel 4.6. kemudian untuk mendapatkan hasil citra tersegmentasi dilakukan dengan mengubah hasil dari morphologi closing tadi ke dalam bentuk biner, nilai piksel yang lebih dari 0 akan diubah menjadi 1, sehingga hasil dari binerisasi hanya berupa nilai 0 atau 1. Hasil akhir dari binerisasi ditunjukkan pada Tabel 4.7.

Tabel 0.5 Nilai piksel *invers*

0	0	0	0	0	0	0	0	0	0
0	0	0	0	255	0	0	0	0	0
0	0	0	255	255	255	255	0	0	0
0	0	255	255	255	255	255	255	0	0
0	0	255	255	255	255	255	255	0	0
0	0	255	255	255	255	255	255	0	0
0	0	0	255	255	255	255	0	0	0
0	0	0	255	255	255	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Tabel 0.6 Nilai piksel morfologi *closing*

0	0	0	0	255	0	0	0	0	0
0	0	0	255	255	255	255	0	0	0
0	0	0	255	255	255	255	0	0	0
0	0	255	255	255	255	255	255	0	0
0	0	255	255	255	255	255	255	0	0
0	0	255	255	255	255	255	255	0	0
0	0	0	255	255	255	255	0	0	0
0	0	0	255	255	255	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Tabel 0.7 Nilai piksel citra tersegmentasi

0	0	0	0	1	0	0	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Selanjutnya hasil akhir dari *pre-processing* yang dilakukan berupa citra RGB yang telah tersegmentasi. Hasil akhir dari citra tersegmentasi tersebut ditampilkan seperti pada Tabel 4.8.

Tabel 0.8 Nilai piksel B/G/R hasil segmentasi

0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	183/ 163/ 150	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0
0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	131/ 151/ 159	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0

0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	133/ 158/ 162	126/ 151/ 158	132/ 161/ 173	132/ 157/ 171	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0
0/ 0/ 0	0/ 0/ 0	141/ 160/ 160	116/ 141/ 149	152/ 169/ 170	67/ 136/ 184	43/ 116/ 182	114/ 145/ 166	0/ 0/ 0	0/ 0/ 0
0/ 0/ 0	0/ 0/ 0	117/ 149/ 146	159/ 174/ 172	152/ 165/ 167	72/ 112/ 170	25/ 91/ 168	86/ 131/ 166	0/ 0/ 0	0/ 0/ 0
0/ 0/ 0	0/ 0/ 0	123/ 141/ 144	161/ 175/ 171	140/ 158/ 160	123/ 140/ 154	52/ 89/ 146	111/ 129 142	0/ 0/ 0	0/ 0/ 0
0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	148/ 160/ 156	160/ 174/ 170	130/ 146/ 147	152/ 161/ 157	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0
0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	93/ 110/ 120	137/ 148/ 143	136/ 146/ 142	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0
0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0
0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0	0/ 0/ 0

4.2.2 Manualisasi HSV

Pada sub bab ini akan dijelaskan mengenai fitur-fitur statistik HSV yang di peroleh berdasarkan perhitugan manual dari rumus-rumus yang ada. Fitur yang akan dihasilkan yaitu *mean*, deviasi standar, *skewness* dan *kurtosis*. Proses mendapatkan fitur-fitur tersebut dijelaskan pada sub bab 4.2.2.2 yang sebelumnya dilakukan konversi citra RGB ke HSV terlebih dahulu pada sub bab 4.2.2.1.

4.2.2.1 Konversi Warna RGB ke HSV

Pada manualisasi, digunakan citra RGB berukuran 3 x 3 yang memiliki 3 dimensi matriks yang dapat dilihat pada Tabel 4.9.

Tabel 0.9 Citra manualisasi RGB

10	19	26
40	52	67
23	23	22
77	92	99
131	141	145
94	99	100
26	29	29
59	68	69
12	15	17

Kemudian citra 3 dimensi tadi diubah menjadi citra dua dimensi dengan memisahkan setiap channelnya seperti Tabel 4.10.

Tabel 0.10 Nilai piksel masing-masing Channel RGB

Channel R			Channel G			Channel B		
26	67	22	19	52	23	10	40	23
99	145	100	92	141	99	77	131	12
29	67	17	29	68	15	26	69	12

Langkah pertama yang dilakukan yaitu mengubah citra RGB menjadi HSV.

Untuk melakukan contoh perhitungan dilakukan dengan mengambil satu nilai piksel pada indeks 0,0 pada setiap *channel*nya, sehingga didapatkan:

B = 10, G = 19, R = 26

Kemudian melakukan normalisasi pada setiap pikselnya yaitu melakukan pembagian dengan nilai 255 seperti pada Persamaan 2.1, 2.2 dan 2.3. Nilai tersebut menjadi:

R = $\frac{26}{255}$ = 0,102 , G = $\frac{19}{255}$ = 0,074, B = $\frac{10}{255}$ = 0,039

Menghitung V menggunakan Persamaan 2.4.

V = max(0,102, 0,074, 0,039)

V = 0,102

Kemudian mengubah nilai V ke 8 bit image menggunakan Persamaan 2.7.

V = 0,102 * 255

V = 26

Menghitung S, Max = R menggunakan Persamaan 2.5.

$$S = \frac{0{,}102 - \min(0{,}102, 0{,}074, 0{,}039)}{0{,}102}$$

$$S = \frac{0{,}102 - 0{,}039}{0{,}102}$$

$$S = \frac{0{,}063}{0{,}102}$$

$$S = 0{,}617$$

Kemudian mengubah nilai S ke 8 bit image menggunakan Persamaan 2.8.

S = 0.617 * 255

S = 157

Menghitung H, V=R menggunakan Persamaan 2.6.

$$H = 60 * \frac{(0{,}074 - 0{,}039)}{0{,}102 - 0{,}039}$$

$$H = 60 * \frac{0{,}035}{0{,}063}$$

$$H = 60 * 0{,}556$$

$$H = 33.333$$

Kemudian mengubah nilai H ke 8 bit image menggunakan persamaan 2.9.

$$H = \frac{33.333}{2}$$

H = 17

Nilai pada channel H pada baris 0 kolom 0 adalah 17 pada channel S pada baris 0 kolom 0 adalah 157, pada *channel* V pada baris 0 kolom 0 adalah 26.

Hasil akhir yang diperoleh dari konversi RGB ke HSV ditampilkan di Tabel 4.11

Tabel 0.11 Nilai piksel hasil konversi RGB ke HSV

Channel H			Channel S			Channel V		
17	13	90	157	103	11	26	67	23
20	21	25	57	25	15	99	145	100
30	27	18	26	37	75	29	69	17

4.2.2.2 *Perhitungan Fitur Statistik HSV*

Kemudian berdasarkan matriks *channel* H diatas perhitungan nilai rata-rata, standar deviasi, *skewness* dan *kurtosis* . Perhitungan tersebut dilakukan sebagai berikut.

- *Mean*

  Untuk mendapatkan nilai mean, *dilakukan* berdasarkan Persamaan 2.10 sehingga mendapatkan hasil sebagai berikut.

  Mean_h = 17 + 13 + 90 + 20 + 21 + 25 + 30 + 27 + 18

  $= \frac{261}{9}$

  = 29

- Deviasi Standar

  Untuk mendapatkan nilai deviasi standar, dilakukan berdasarkan Persamaan 2.11 sehingga mendapatkan hasil sebagai berikut.

  $$Sd\_h = \sqrt{\frac{(17-29)^2 + (13-29)^2 + (90-29)^2 + (20-29)^2 + (25-29)^2 + (30-29)^2 + (27-29)^2 + (18-29)^2}{}}$$

  $$= \sqrt{\frac{64 + 256 + 3.721 + 81 + 64 + 16 + 1 + 4 + 121}{9}}$$

  $$= \sqrt{\frac{4328}{9}}$$

  $$= \sqrt{480{,}888}$$

  = 21,929

- *Skewness*

  Untuk mendapatkan nilai skewness, dilakukan berdasarkan Persamaan 2.12 sehingga mendapatkan hasil sebagai berikut.

  Skew_h = $(17-29)^3 + (13-29)^3 + (90-29)^3 + (20-29)^3 + (21-29)^3 + (25-29)^3 + (30-29)^3 + (27-29)^3 + (18-29)^3$

  =-(1.728)+-(4.096)+-(226.981)+-(729)+-(512)+-(64)+1+-(8)+-(1.331)

  $$= \frac{-(235.228)}{9 * 21{,}929^3}$$

  $$= \frac{-(235.228)}{94907{,}163132}$$

  = -2,478

- *Kurtosis*

  Untuk mendapatkan nilai *kurtosis*, dilakukan berdasarkan Persamaan 2.13 sehingga mendapatkan hasil sebagai berikut.

Kurtosis_h = $(17-29)^4 + (13-29) + (90-29)^4 + (20-29)^4 + (21-29)^4 + (25-29)^4 + (30-29)^4 + (27-29)^4 + (18-29)^4$

= 20.736+65.536+13.845.841+6.561+ 4.096 + 256 + 1 + 16 + 14.641

$= \frac{13.957.684}{9 * (21,929)^4} - 3$

$= \frac{13.957.684}{231.246.575.593.243.681} - 3$

$= \frac{13.957.684}{231.246.575.593.243.678} - 3$

= 6.038446

4.2.3 Manualisasi GLCM

Pada sub bab ini akan dijelaskan mengenai fitur-fitur statistik GLCM yang di peroleh berdasarkan perhitugan manual dari rumus-rumus yang ada. Fitur yang akan dihasilkan yaitu *meanX, meanY, varianceX, varianceY, energy, entropy, contrast, dissimilarity, homogeneity dan correlation* dari empat sudut. Proses mendapatkan fitur-fitur tersebut dijelaskan pada sub bab 4.3.3.3 yang sebelumnya dilakukan perhitungan matriks *Co – Occurrence* terlebih dahulu pada sub bab 4.3.3.1 dan normalisasi matriks *Co – Occurrence* pada sub bab 5.3.3.2.

4.2.3.1 Pembentukan Co – Occurrence Matrix

Pada perhitungan manual untuk mendapatkan fitur-fitur GLCM, akan dilakukan dengan membuat matriks *gray* level berukuran 4 x 3 sebagai sampel. Matriks yang digunakan ditunjukkan pada Tabel 4.12.

Tabel 0.12 Sampel nilai Matriks untuk manualisasi GLCM

citraGray =	1	1	3	4
	0	0	2	4
	2	4	3	1

Setelah membuat matriks, langkah selanjutnya yaitu menghitung ukuran matriks (ukCM) menggunakan Persamaan 2.14.

ukCM = 4 - 0 + 1 = 5

Jadi, ukuran matriks *Co – Occurrence* adalah 5 * 5

Setelah mendapatkan ukuran matriks *Co - Occurrence*, langkah selanjutnya adalah membentuk matriks *Co – Occurrence* berdasarkan matriks citraGray, arah sudut dan d.

Pembentukan matriks *Co - ocurence* dengan sudut 0

CM untuk 0 derajat, dimana d = {0, 1}

Tabel 0.13 Matriks Co – Occurrence 0 derajat

i/j	0	1	2	3	4
0	1	0	1	0	0
1	0	1	0	1	0
2	0	0	0	0	2
3	0	1	0	0	1
4	0	0	0	1	0

CM untuk 45 derajat, dimana d = {-1, 1}

Tabel 0.14 Matriks Co – Occurrence 45 derajat

i/j	0	1	2	3	4
0	0	1	0	1	0
1	0	0	0	0	0
2	1	0	0	0	1
3	0	0	0	0	1
4	0	0	1	0	0

CM untuk 90 derajat, dimana d = {-1, 0}

Tabel 0.15 Matriks Co – Occurrence 90 derajat

i/j	0	1	2	3	4
0	0	2	0	0	0
1	0	0	0	0	1
2	1	0	0	1	0
3	0	0	1	0	0
4	1	0	0	0	1

CM untuk 135 derajat, dimana d = {-1, -1}

Tabel 0.16 Matriks Co – Occurrence 135 derajat

i/j	0	1	2	3	4
0	0	1	0	0	0
1	0	0	1	0	0
2	0	1	0	0	0
3	1	0	0	0	0
4	1	0	0	1	0

Kemudian untuk melanjutkan manualisasi, matriks yang digunakan yaitu matriks CM 0 derajat pada seperti pada Tabel 4.13.

4.2.3.2 Normalisasi Co – Occurrence Matrix

Kemudian nilai elemen GLCM perlu di normalisasi sehingga jumlahnya bernilai 1. Dengan demikian, akan menjadi seperti Tabel 4.17.

Tabel 0.17 Normalisasi matriks CM 0 derajat

$\frac{1}{9}$	$\frac{0}{9}$	$\frac{1}{9}$	$\frac{0}{9}$	$\frac{0}{9}$
$\frac{0}{9}$	$\frac{1}{9}$	$\frac{0}{9}$	$\frac{1}{9}$	$\frac{0}{9}$
$\frac{0}{9}$	$\frac{0}{9}$	$\frac{0}{9}$	$\frac{0}{9}$	$\frac{2}{9}$
$\frac{0}{9}$	$\frac{1}{9}$	$\frac{0}{9}$	$\frac{0}{9}$	$\frac{1}{9}$
$\frac{0}{9}$	$\frac{0}{9}$	$\frac{0}{9}$	$\frac{1}{9}$	$\frac{0}{9}$

Dengan demikian didapatkan nilai citra GLCM ternormalisasi seperti berikut Tabel 4.18.

Tabel 0.18 Hasil normalisasi matriks CM 0 derajat

0,111	0	0,111	0	0
0	0,111	0	0,111	0
0	0	0	0	0,222
0	0,111	0	0	0,111
0	0	0	0,111	0

4.2.3.3 Menghitung Fitur Statistik GLCM

Setelah mendapatkan matriks GLCM yang telah dinormalisasi, langkah selanjutnya adalah menghitung fitur statistik GLCM, perhitungannya dilakukan sebagai berikut:

- *Mean X*

  Untuk mendapatkan nilai *mean X*, dilakukan berdasarkan Persamaan 2.15 sehingga mendapatkan hasil sebagai berikut

  Mean X = 0 * 0,111 + 0 * 0 + 0 * 0,111 + 0 * 0 + 0 * 0 + 1 * 0 + 1 * 0,111 + 1* 0 +1 * 0,111 + 1* 0 + 2* 0 + 2* 0 + 2* 0 + 2* 0 + 2* 0,222 + 3* 0 + 3* 0,111 + 3* 0 + 3* 0 + 3* 0,111 + 4 * 0 + 4* 0 + 4* 0 + 4* 0,111 + 4* 0

= 0,111 + 0,111 + 0,444 + 0,333 + 0,333 + 0,444

= 1,776

- *Mean Y*

  Untuk mendapatkan nilai *mean Y*, dilakukan berdasarkan Persamaan 2.16 sehingga mendapatkan hasil sebagai berikut:

  Mean Y = 0 * 0,111 + 1 * 0 + 2 * 0,111 + 3 * 0 + 4 * 0 + 0 * 0 + 1 * 0,111 + 2* 0 + 3 * 0,111 + 4* 0 + 0* 0 + 1* 0 + 2* 0 + 3* 0 + 4* 0,222 + 0* 0 + 1* 0,111 + 2* 0 + 3* 0 + 4* 0,111 + 0* 0 + 1* 0 + 2* 0 + 3* 0,111 + 4* 0

  = 0,222 + 0,111 + 0,333 + 0,888 + 0,111 + 0,444

  = 2,109

- *Variance X*

  Untuk mendapatkan nilai *variance X*, dilakukan berdasarkan Persamaan 2.17 sehingga mendapatkan hasil sebagai berikut:

  $VarX = (0-1,776)^2 * 0,111 + (0-1,776)^2 * 0 + (0-1,776)^2 * 0,111 + (0-1,776)^2 * 0 + (0-1,776)^2 * 0 + (1-1,776)^2 * 0 + (1-1,776)^2 * 0,111 + (1-1,776)^2 * 0 + (1-1,776)^2 * 0,111 + (1-1,776)^2 * 0 + (2-1,776)^2 * 0 + (2-1,776)^2 * 0 + (2-1,776)^2 * 0 + (2-1,776)^2 * 0 + (2-1,776)^2 * 0,222 + 3-1,776)^2 * 0 + (3-1,776)^2 * 0,111 + (3-1,776)^2 * 0 + (3-1,776)^2 * 0 + (3-1,776)^2 * 0,111 + (4-1,776)^2 * 0 + (4-1,776)^2 * 0 + (4-1,776)^2 * 0 + (4-1,776)^2 * 0,111 + (4-1,776)^2 * 0$

  = 0,350 + 0,350 + 0,067 + 0,067 + 0,0111 + 0,926 + 0,926 + 0.549

  = 2,420

- *Variance Y*

  Untuk mendapatkan nilai *variance Y*, dilakukan berdasarkan Persamaan 2.18 sehingga mendapatkan hasil sebagai berikut:

  $VarY = (0-1,776)^2 * 0,111 + (1-1,776)^2 * 0 + (2-1,776)^2 * 0,111 + (3-1,776)^2 * 0 + (4-1,776)^2 * 0 + (0-1,776)^2 * 0 + (1-1,776)^2 * 0,111 + (2-1,776)^2 * 0 + (3-1,776)^2 * 0,111 + (4-1,776)^2 * 0 + (0-1,776)^2 * 0 + (1-1,776)^2 * 0 + (2-1,776)^2 * 0 + (3-1,776)^2 * 0 + (4-1,776)^2 * 0,222 + (0-1,776)^2 * 0 + (1-1,776)^2 * 0,111 + (2-1,776)^2 * 0 + (3-1,776)^2 * 0 + (4-1,776)^2 * 0,111 + (0-1,776)^2 * 0 + (1-1,776)^2 * 0 + (2-1,776)^2 * 0 + (3-1,776)^2 * 0,111 + (4-1,776)^2 * 0$

  = 0,350 + 0.006 + 0,067 + 0,926 + 1,098 + 0,549 + 0,926

  = 3.922

- *Energy*

  Untuk mendapatkan nilai *energy*, dilakukan berdasarkan Persamaan 2.19 sehingga mendapatkan hasil sebagai berikut:

Energy = $(0,111)^2+(0,111)^2+(0,111)^2 + (0,111)^2 + (0,222)^2 + (0,111)^2 + (0,111)^2 + (0,111)^2$

= 0,012 + 0,012 + 0,012 + 0,012 + 0,049 + 0,012 + 0,012 + 0,012

= 0,133

- *Entropy*

  Untuk mendapatkan nilai *entropy*, dilakukan berdasarkan Persamaan 2.20 sehingga mendapatkan hasil sebagai berikut:

  Entropy = -(0,111 x log (0,111)) + -(0,111 x log (0,111)) + - (0,111 x log (0,111)) + -(0,111 x log (0,111)) + -(0,222 x log (0,222)) + -(0,111 x log (0,111)) + -(0,111 x log (0,111)) + -(0,111 x log (0,111))

  = -(0,111 x (-0,955)) + -(0,111 x (-0,955)) + - (0,111 x (-0,955)) + -(0,111 x (-0,955)) + -(0,222 x (-0,653)) + -(0,111 x (-0,955)) + -(0,111 x (-0,955)) + -(0,111 x (-0,955))

  =-(-0,106)+-(-0,106)-(-0,106)+-(-0,106)+-(-0,145)+-(-0,106)+-(-0,106)+-(0,106)

  = 0,887

- *Contrast*

  Untuk mendapatkan nilai *contrast*, dilakukan berdasarkan Persamaan 2.21 sehingga mendapatkan hasil sebagai berikut:

  Contras t = $((0 - 0)^2$ x 0,111) + $((0 - 2)^2$ x 0,111) + $((1 - 1)^2$ x 0,111) + $((1 - 3)^2$ x 0,111) + $((2 - 4)^2$ x 0,222) + $((3 – 1)^2$ x 0,111) + $((3 - 4)^2$ x 0,111) + $((4 - 3)^2$ x 0,111)

  = 4 x 0,111 + 4 x 0,111 + 4 x 0,222 + 4 x 0,111 + 1 x 0,111 + 1 x 0,111

  = 0,444 + 0,444 + 0,888 + 0,444 + 0,111 + 0,111

  = 2,442

- *Dissimilarity*

  Untuk mendapatkan nilai *dissimilarity*, dilakukan berdasarkan Persamaan 2.22 sehingga mendapatkan hasil sebagai berikut:

  Dissimilarity = (|0 – 0 | x 0,111) + (|0 - 2 | x 0,111) + (|1 - 1 | x 0,111) + (|1 - 3 | x 0,111) + (|2 - 4 | x 0,222) + (|3 – 1 | x 0,111) + (|3 - 4 | x 0,111) + (|4 - 3 | x 0,111)

  = (0 x 0,111) + (2 x 0,111) + (0 x 0,111) + (2 x 0,111) + (2 x 0,222) + (2 x 0,111) + (1 x 0,111) + (1 x 0,111)

  = 0 + 0,222 + 0 + 0,222 + 0,444 + 0,222 + 0,111 + 0,111

  = 1,11

- *Homogenity*

  Untuk mendapatkan nilai *homogeneity*, dilakukan berdasarkan Persamaan 2.23 sehingga mendapatkan hasil sebagai berikut:

$$Homogenity = \frac{0{,}111}{1 + |0-0|} + \frac{0{,}111}{1 + |0-2|} + \frac{0{,}111}{1 + |1-1|} + \frac{0{,}111}{1 + |1-3|} + \frac{0{,}222}{1 + |2-4|} + \frac{0{,}111}{1 + |3-1|} + \frac{0{,}111}{1 + |3-4|} + \frac{0{,}111}{1 + |4-3|}$$

$$= 0{,}111 + 0{,}037 + 0{,}111 + 0{,}037 + 0{,}019 + 0{,}037 + 0{,}056 + 0{,}056$$

$$= 0{,}464$$

- *Correlation*

  Untuk mendapatkan nilai *Correlation*, dilakukan berdasarkan Persamaan 2.24 sehingga mendapatkan hasil sebagai berikut:

  MeanY = 2,109

  SdX = $\sqrt{2{,}420}$ = 1,555

  SdY = $\sqrt{3{,}922}$ = 1,980

  Correlation = ((0-2,109) * (0-2,109) * 0,111) / (1,556 * 1,980) + ((0-2,109) *(1-2,109) * 0) / (1,556 * 1,980) + ((0-2,109) *(2-2,109) * 0,111) / (1,556 * 1,980) + ((0-2,109) *(3-2,109)* 0) / (1,556 * 1,980) +((0-2,109) *(4-2,109)* 0) / (1,556 * 1,980) + ((1 -2,109)* (0 -2,109)* 0) / (1,556 * 1,980) + ((1-2,109) * (1-2,109)* 0,111) / (1,556 * 1,980) + ((1-2,109)* (2-2,109) * 0) / (1,556 * 1,980) + ((1-2,109) * (3-2,109) * 0,111) / (1,556 * 1,980) + ((1-2,109)* (4-2,109)* 0) / (1,556 * 1,980) + ((2-2,109)* (0-2,109)* 0) / (1,556 * 1,980) + ((2-2,109)* (1-2,109)* 0) / (1,556 * 1,980) + ((2-2,109)* (2-2,109)* 0) / (1,556 * 1,980) +((2-2,109)* (3-2,109)* 0) / (1,556 * 1,980) + ((2-2,109)* (4-2,109)* 0,222) / (1,556 * 1,980) + ((3-2,109)* (0-2,109)* 0) / (1,556 * 1,980) + ((3-2,109)* (1-2,109)* 0,111) / (1,556 * 1,980) + ((3-2,109)* (2-2,109)* 0) / (1,556 * 1,980) + ((3-2,109)* (3-2,109)* 0) / (1,556 * 1,980) + ((3-2,109)* (4-2,109)* 0,111) / (1,556 * 1,980) + ((4-2,109) *(0-2,109)* 0) / (1,556 * 1,980) + ((4-2,109) *(1-2,109)* 0) / (1,556 * 1,980) + ((4-2,109) *(2-2,109)* 0) / (1,556 * 1,980) + ((4-2,109) * (3-2,109)* 0,111) / (1,556 * 1,980) + ((4-2,109) * (4-2,109)* 0) / (1,556 * 1,980)

  = 0,4524 / 3,080 + 0,025 / 3,080 + 0,1365 / 3,080 + -0,109 / 3,080 + -0,045 / 3,080 + -0,109 / 3,080 + 0.1870 / 3,080 + 0,1870 / 3,080

  = 0,1468 + 0,0081 + 0,0443 + -0,0353 + -0,0146 + -0,0353 + 0,0607 + 0,0607

  = 0.3206 – 0,0852

  = 0,2354

4.2.4 Manualisasi Information Gain

Pada manualisasi, dilakukan contoh perhitungan manual terhadap tiga kelas makanan yaitu kelas 0, 1 dan 2 yang masing-masing kelasnya memiliki empat fitur yaitu mean_h, skewness_s, dissimilarity0 dan entropy45. Data yang digunakan dapat dilihat pada Tabel 4.19.

Tabel 0.19 Data perhitungan manual *Information Gain*

mean_h	skewness_s	dissim0	entropy45	kelas
6.885664	1.083143332	1.2619	1.3601	0
7.007252	1.077075243	1.215864	1.362087	0
11.41084	-0.22557985	4.671651	2.666295	0
13.71196	4.27935441	11.38654	2.949972	1
12.94906	4.119089087	11.23971	2.71326	1
17.41266	5.19354229	16.52202	3.795062	1
14.92346	2.218662674	7.635102	2.713729	2
13.73996	2.037824119	8.258028	2.435331	2
20.68532	2.620360649	11.9847	3.440213	2

4.2.4.1 Mengubah Data Numerik Menjadi Kategori

Perhitungan Information Gain membutuhkan data kategori, untuk itu langkah pertama yang harus dilakukan yaitu melakukan pengubahan data numerik menjadi data kategori. Manualisasi mengubah data menjadi kategori dilakukan pada fitur mean_h. Data mean_h yang akan dirubah menjadi kategori ditunjukkan pada Tabel 4.20.

Tabel 0.20 Fitur mean_h untuk perhitungan kategorisasi data

mean_h
6.885664
7.007252
11.41084
13.71196
12.94906
17.41266
14.92346
13.73996
20.68532

Langkah selanjutnya yaitu mencari nilai minimum, maksimum, jangkauan banyak kelas dan panjang kelas berdasarkan persamaan 2.25 sampai 2.27.

Minimum = min (6.885664, 7.007252, 11.41084, 13.71196, 12.94906, 17.41266, 14.92346, 13.73996, 20.68532) = 6.885664

Maksimum = max (6.885664, 7.007252, 11.41084, 13.71196, 12.94906, 17.41266, 14.92346, 13.73996, 20.68532) = 20.685316

Untuk menghitung J dilakukan bedasarkan Persamaan 2.25.

Jangkauan = Maksimum – Minimum = 13.799652

Untuk menghitung K dilakukan bedasarkan Persamaan 2.26.

Banyak kelas = 1 + 3.3 log (9) = 4.149000281 ≈ 4

Untuk menghitung interval dilakukan bedasarkan Persamaan 2.27.

Interval = jangkauan/banyak kelas = 13.799652/ 4 = 3.449913

Kemudian memasukkan setiap datanya ke dalam kategori berdasarkan interval dari batas bawah dan batas atas. Batas bawah pada kategori pertama merupakan nilai terkecil, sedangkan batas atas merupakan hasil penjumlahan dari batas bawah dengan interval. Untuk batas atas dan batas bawah pada kategori selanjutnya merupakan nilai dari batas atas kategori sebelumnya sedangkan batas atasnya masih berupa penambahan antara batas bawah dengan interval. Contoh memasukkan setiap data pada mean_h ke kategorinya ditunjukkan pada Tabel 4.21

Tabel 0.21 Memasukkan data ke kategori

Batas bawah	Batas atas	Data mean_h	Data kategori mean_h
6.885664	10.335577	6.885664	0
10.335577	13.785490000000001	7.007252	0
13.785490000000001	17.235403	11.41084	1
17.235403	20.685316	13.71196	1
		12.94906	1
		17.41266	3
		14.92346	2
		13.73996	1
		20.68532	3

Begitu juga dengan skewness_s, dissimilarity0 dan entropy45. Setelah melakukan pengubahan data tunggal menjadi data berkelompok yang telah dimasukkan ke katogori maka didapatkan hasil seperti pada Tabel 4.22.

Tabel 0.22 Hasil pengubahan data ke kategori

mean_h	skewness_s	Gdissim0	Gentropy45	kelas
0	0	0	0	0
0	0	0	0	0
1	0	0	2	0
1	3	2	2	1
1	3	2	2	1
3	3	3	3	1
2	1	1	2	2
1	1	1	1	2
3	2	2	3	2

4.2.4.2 Menghitung Entropy

Perhitungan nilai entropy dilakukan pada setiap kelas, perhitungan tersebut dilakukan berdasarkan Persamaan 2.28 sehingga mendapatkan hasil sebagai berikut:

$$Entropy(S) = (-P001\ log_2 P001) + (-P002\ log_2 P002) + (-P004\ log_2\ P004)$$

$$= \left(-\frac{3}{9} log_2 \frac{3}{9}\right) + \left(-\frac{3}{9} log_2 \frac{3}{9}\right) + \left(-\frac{3}{9} log_2 \frac{3}{9}\right)$$

= 0.528321 + 0.528321 + 0.528321

= 1.584963

4.2.4.3 Menghitung Gain Setiap Fitur

Sebelum melakukan perhitungan *Gain*, terlebih dahulu dilakukan perhitungan nilai entropy berdasarkan atribut dengan kemudian menghitung *Gain* fitur menggunakan Persamaan 2.29.

a. Kategori 0

$$Entropy(S0) = (-P0 log_2 P0) + (-P1 log_2 P1) + (-P2 log_2 P2)$$

$$= \left(-\frac{2}{2+0+0} log_2 \frac{2}{2+0+0}\right) + \left(-\frac{0}{2+0+0} log_2 \frac{0}{2+0+0}\right) + \left(-\frac{0}{2+0+0} log_2 \frac{0}{2+0+0}\right)$$

= 0 + 0 + 0

= 0

b. Kategori 1

$$Entropy(S1) = (-P0 log_2 P0) + (-P1 log_2 P1) + (-P2 log_2 P2)$$

$$= \left(-\frac{1}{1+2+1} log_2 \frac{1}{1+2+1}\right) + \left(-\frac{2}{1+2+1} log_2 \frac{2}{1+2+1}\right) + \left(-\frac{1}{1+2+1} log_2 \frac{1}{1+2+1}\right)$$

= 0.5 + 0.5 + 0.5

= 1.5

c. Kategori 2

$$Entropy(S2) = (-P0log_2P0) + (-P1log_2P1) + (-P2log_2P2)$$

$$= \left(-\frac{0}{0+0+1}log_2\frac{0}{0+0+1}\right) + \left(-\frac{0}{0+0+1}log_2\frac{0}{0+0+1}\right) + \left(-\frac{1}{0+0+1}log_2\frac{1}{0+0+1}\right)$$

= 0 + 0 + 0

= 0

d. Kategori 3

$$Entropy(S3) = (-P0log_2P0) + (-P1log_2P1) + (-P2log_2P2)$$

$$= \left(-\frac{0}{0+1+1}log_2\frac{0}{0+1+1}\right) + \left(-\frac{1}{0+1+1}log_2\frac{1}{0+1+1}\right) + \left(-\frac{1}{0+1+1}log_2\frac{1}{0+1+1}\right)$$

= 0 + 0.5 + 0.5

= 1

Selanjutnya dilakukan perhitungan nilai Gain pada fitur mean_h menggunakan Persamaan 2.29.

$$Gain(S, me\_h) = 1.584963 - \left\{\left(\left(\frac{2+0+0}{9}\right)\text{x } 0\right) + \left(\left(\frac{1+2+1}{9}\right)\text{x } 1.5\right) + \left(\left(\frac{0+0+1}{9}\right)\text{x } 0\right) + \left(\left(\frac{0+1+1}{9}\right)\text{x } 1\right)\right\}$$

$$= 1.584963 - \{0 + 0{,}667 + 0 + 0{,}222\}$$

$$= 1.584963 - 0{,}889$$

$$= 0.696074$$

Sehingga di dapatkan nilai Gain mean_h yaitu 0.69593.

Dengan perhitungan yang sama dilakukan perhitungan gain pada skewness_s, dissimilarity_0 dan entropy_45, sehingga di dapatkan hasil sebagai berikut:

$Gain(S, \text{skewness\_s})$ = 1.584963

$Gain(S, \text{dissimilarity\_0})$ = 1.278864

$Gain(S, \text{entropy\_45})$ = 0.69607

4.2.4.4 Mengurutkan Nilai Gain dan Mendapatkan Hasil Seleksi Fitur

Tabel 0.23 Mengurutkan nilai gain dari yang terbesar

No.	Fitur	Nilai Gain
1	skewness_s	1.584963
2	dissimilarity_0	1.278864
3	mean_h	0.696074
4	Entropy_45	0.696074

Kemudian menentukan berapa fitur yang akan diambil, pada manualisasi ini ingin mengambil 2 fitur untuk dilanjutkan ke proses klasifikasi, maka fitur yang digunakan adalah skewness_s dan dissimilarity_0.

4.2.5 Manualisasi K – Nearest Neighbor

Proses klasifikasi dilakukan berdasarkan hasil seleksi fitur information gain, fitur-fitur yang didapatkan yaitu skewness_s dan Gdissim0. Data latih yang digunakan untuk manualisasi KNN dapat dilihat pada Tabel 4.24 dan data ujinya terdapat pada Tabel 4.25.

Tabel 0.24 Data latih KNN

Data ke	skewness_s	Gdissim0	kelas
1	1,083143332	1,2619	0
2	1,077075243	1,215864	0
3	-0,22557985	4,671651	0
4	4,27935441	11,38654	1
5	4,119089087	11,23971	1
6	5,19354229	16,52202	1
7	2,218662674	7,635102	2
8	2,037824119	8,258028	2
9	2,620360649	11,9847	2

Tabel 0.25 Data uji KNN

Data ke	skewness_s	Gdissim0	kelas
10	6,695404	1,263575	?
11	12,70291	11,18578	?
12	15,06521	7,613595	?

Sebelum menghitung jarak, dilakukan normalisasi data latih dan data uji untuk menyetarakan rentang data pada setiap fitur menggunakan Persamaan 2.31. Hasil normalisasi data latih dan data uji di tampilkan pada Tabel 4.26 dan 4.27.

Tabel 0.26 Data Latih KNN Ternormalisasi

Data ke	skewness_s	Gdissim0	kelas
1	0,07814392	0,09003	0
2	0,0777816	0,087244	0
3	0	0,296406	0
4	0,26898984	0,702827	1
5	0,25942039	0,69394	1
6	0,32357605	1,013653	1
7	0,14594584	0,47577	2
8	0,13514796	0,513473	2
9	0,16993125	0,739031	2

Tabel 0.27 Data Uji KNN Ternormalisasi

Data ke	skewness_s	Gdissim0	kelas
1	6,695404	1,263575	?
2	12,70291	11,18578	?
3	15,06521	7,613595	?

Langkah selanjutnya yaitu melakukan perhitungan jarak seperti pada Persamaan 2.30. Contoh perhitungan jarak menggunakan Euclidean distance.

$$d_{(10,1)=}\sqrt{(6{,}695404 - 0{,}07814392)^{\,2} + (1{,}263575 - 0{,}09003)^{\,2}}$$

$$d_{(10,1)} = 0.79202868$$

Hasil dari perhitungan jarak Euclidean antara data uji 1 dengan data latih secara manual dapat dilihat pada Tabel 4.28.

Tabel 0.28 Hasil perhitungan jarak

	Jarak	Klasifikasi
D(10, 1)	0,79202868	0
D(10, 2)	0,79475865	0
D(10, 3)	0,6447258	0
D(10, 4)	0,1575608	1
D(10, 5)	0,17042868	1
D(10, 6)	0,18909319	1
D(10, 7)	0,41475312	2
D(10, 8)	0,38940307	2
D(10, 9)	0,21345593	2

Langkah berikutnya yaitu mengambil data dengan nilai yang paling dekat dengan tetangga. Nilai K yang digunakan pada manualisasi ini adalah 3, sehingga diambil 3 data teratas dari hasil pengurutan jarak. Tabel 4.29 menunjukkan hasil dari pemilihan data sebanyak K.

Tabel 0.29 Hasil pemilihan tetangga sebanyak K

	Jarak	klasifikasi
D(10, 4)	0,1575608	1
D(10, 5)	0,17042868	1
D(10, 9)	0,21345593	2

Kemudian dipilih kelas mayor, yaitu kelas yang paling dominan dari hasil pemilihan data sebanyak *k*. berdasarkan Tabel 4.34 kelas mayor yang didapatkan adalah kelas 1 yaitu jenis makanan Roti Gandum.

4.2.6 Manualisasi Akurasi

Akurasi dapat dihitung menggunakan Persamaan 2.30.

Diketahui jumlah data uji = 105, Jumlah data uji yang benar diklasifikasikan = 89

$$akurasi = \frac{89}{105} x\ 100\ \%$$

= 84,76 %

4.3 Perancangan Pengujian

Pengujian dilakukan untuk mengetahui tingkat akurasi sistem yang dapat dilakukan dengan menggunakan Persamaan 2.28. Penelitian ini juga melakukan tiga macam pengujian algoritme yaitu ekstraksi fitur dengan melakukan kombinasi fitur yang digunakan untuk klasifikasi, data yang digunakan pada klasifikasi baik yang menggunakan seleksi fitur maupun tanpa menggunakan seleksi fitur dan pengaruh penggunaan nilai K pada KNN. Perancangan pengujian dapat dilihat pada Tabel 4.30.

Tabel 0.30 Perancangan pengujian

Nilai K	GLCM dengan seleksi fitur	HSV dengan seleksi fitur	GLCM+ HSV dengan seleksi fitur	GLCM tanpa seleksi fitur	HSV tanpa seleksi fitur	GLCM+ HSV tanpa seleksi fitur
3						
5						
7						
9						
11						
13						
15						

BAB 5 IMPLEMENTASI

Pada bab ini, dijelaskan mengenai implementasi sistem berdasarkan perancangan yang telah disusun pada bab empat. Bab ini terdiri dari lingkungan untuk melakukan implementasi sistem, batasan implementasi dan implementasi sistem. Sub bab lingkungan implementasi berisi penjelasan mengenai spesifikasi kebutuhan yang digunakan pada penelitian ini berupa kebutuhan software dan hardware. Sub bab batasan implementasi berisi batasan-batasan yang diterapkan pada penelitian. Sub bab implementasi berisi implementasi berupa kode program dan penjelasannya.

5.1 Lingkungan Implementasi

Pada sub bab ini berisi penjelasan mengenai lingkungan implementasi sistem Seleksi fitur Information Gain pada klasifikasi citra makanan berdasarkan HSV dan GLCM yang terdiri dari lingkungan *hardware* dan lingkungan *software*.

5.1.1 Lingkungan Hardware

Lingkungan hardware yang digunakan dalam pembuatan sistem Seleksi fitur Information Gain pada klasifikasi citra makanan berdasarkan HSV dan GLCM adalah sebagai berikut.

1. Preprocessor: Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.71 GHz
2. Memory: 4,00 GB
3. Hardisk: 1 TB
4. Monitor: 14 inch
5. Keyboard: -
6. Mouse: Genius M/N:NetScroll 120

5.1.2 Lingkungan Software

Lingkungan hardware yang digunakan dalam pembuatan sistem Seleksi fitur Information Gain pada klasifikasi citra makanan berdasarkan HSV dan GLCM adalah sebagai berikut.

1. Operating System Microsoft Windows 10.
2. Editor pemrograman Python 3.6 dengan paket pendukung sistem menggunakan *cv2, numpy, math, csv* dan *pandas*.
3. Microsoft Office word 2016 yang digunakan untuk membuat laporan penelitian.
4. Microsoft Office Excell 2016 yang digunakan untuk melakukan penyimpanan data.
5. Microsoft Power Point 2016 yang digunakan untuk membuat presentasi penelitian.

5.2 Implementasi Sistem

Pada sub bab ini, dijelaskan setiap proses pada implementasi metode-metode yang digunakan dalam penelitian ini. Proses tersebut meliputi pre-processing, ekstraksi warna HSV, ekstraksi tekstur GLCM, *Information Gain, K – Nearest Neighbor* dan akurasi sistem.

5.2.1 Implementasi Pre-Processing

Inti dari proses *pre-processing* pada penelitian ini yaitu mendapatkan citra tersegmentasi. Terdapat tahapan-tahapan yang dilakukan untuk mendapatkan citra tersebut, meliputi filtering, konversi citra RGB ke LAB, *bitwise, dilasi dan erosi*. Proses *pre-processing* secara lengkap dapat dilihat pada Algoroitme 1.

Algoritme 1: Pre-processing citra

```
1   for i in range (0,len(data)):
2       for files in glob.glob(data[i] + "*"):
3           citra = cv2.imread(files)
4           cresize = cv2.resize(citra, (500, 500))
5           mask = cv2.GaussianBlur(cresize, (11, 11), 0)
6           lab = cv2.cvtColor(mask, cv2.COLOR_BGR2LAB)
7           b = lab[:,:,2]
8           thresh,th3=
            cv2.threshold(b,0,255,cv2.THRESH_BINARY_INV+cv2.THRESH_
            OTSU)
9           inv = cv2.bitwise_not(thresh)
10          kernel = np.ones((11,11), np.uint8)
11          dilasi = cv2.dilate(inv,kernel,iterations = 1)
12          erosi = cv2.erode(dilasi,kernel,iterations = 1)
13          erosi[erosi>0]=1
14          h, w = cresize.shape[:2]
15          for y in range(h):
16              for x in range(w):
17                  if (erosi[y][x] == 0):
18                      cresize[y][x] = 0
19          cv2.imwrite(files+"_baru.jpg", cresize)
```

Penjelasan:
baris 1-2: digunakan untuk melakukan perulangan pada setiap data ke-i sampai dengan data terakhir, dimana setiap datanya melakukan proses dari baris 10-26.
baris 3: membaca data pada indeks ke i.
baris 4: melakukan resize citra dengan ukuran 500 x 500.
baris 5: melakukan filtering Gaussian Blur menggunakan kernel 11 x 11.
baris 6: mengkonversi citra RGB menjadi citra LAB.
baris 7: mengambil channel b dari citra LAB.
baris 8: melakukan thresholding binary menggunakan metode otsu melalui opencv.
baris 9: melakukan invers menggunakan fungsi bitwise malalui opencv.
baris 10: menginisialisasi kernel dengan ukuran 11 x 11.
baris 11: melakukan proses dilasi melalui opencv.
baris 12: melakukan proses erosi melalui opencv.
baris 13-18: digunakan untuk melakukan binerisasi berdasarkan hasil erosi yaitu dengan cara merubah nilai yang bernilai lebih dari nol menjadi bernilai 1.
baris 19: untuk membuat file berdasarkan hasil dari segmentasi.

5.2.2 Implementasi Ekstraksi Fitur Warna

Ekstraksi fitur warna digunakan untuk mendapatkan nilai fitur statistik warna yang pada penelitian ini menggunakan *Color Moment HSV*. Citra yang digunakan pada proses ini berupa citra HSV, sehingga terlebih dahulu dilakukan konversi warna dari citra RGB ke HSV kemudian dihitung fitur statistiknya pada setiap *channel* HSV.

5.2.2.1 Konversi Warna RGB ke HSV

Konversi warna RGB ke HSV digunakan untuk merubah citra yang memiliki ruang warna RGB menjadi citra yang memiliki ruang warna HSV. Hal ini dilakukan karena penelitian ini menggunakan *Color Momen* HSV. Proses konversi dapat dilihat pada Algoritme 2.

Algoritme 2: Konversi Warna RGB ke HSV

```
1   def rgb_to_hsv(b, g, r):
2       for i in range (0,len(b)):
3           for j in range (0,len(b[0])):
4               bz[i][j] = b[i][j]/255.0
5               gz[i][j] = g[i][j]/255.0
6               rz[i][j] = r[i][j]/255.0
7               Maks = np.amax([bz[i][j], gz[i][j], rz[i][j]])
8               Min = np.amin([bz[i][j], gz[i][j], rz[i][j]])
9               delta = Maks-Min
10              if Maks == Min:
11                  H[i][j]=0
12              elif Maks == rz[i][j]:
13                  H[i][j] == (60*(gz[i][j]-bz[i])/delta)
14              elif Maks == gz[i][j]:
15                  H[i][j] == (120 + 60*(bz[i][j]-rz[i][j])/ delta)
                elif Maks == bz[i][j]:
16                  H[i][j] = (240 + 60*(rz[i][j]-gz[i][j])/delta)
17              if Maks == 0:
18                  S[i][j] == 0
19              else:
20                  S[i][j] = (delta/Maks)*255
21              V[i][j] = Maks*255
22              H[i][j] /= 2
23      gethsv = cv2.merge((H, S, V))
24      gethsv = gethsv.astype(np.uint8)
25      return gethsv
```

Penjelasan:

baris 2 – 3: merupakan perulangan perulangan untuk setiap piksel matriks pada setiap channelnya. Pada setiap piksel *channel* melakukan proses pada baris 4 – 23.

baris 4-6: digunakan untuk melakukan normalisasi piksel.

baris 7: digunakan untuk mencari nilai maksimum dari seluruh channel indeks ke i.

baris 8: digunakan untuk mencari nilai minimum dari seluruh channel indeks ke i.

baris 9: digunakan untuk menncari nilai delta, yaitu hasil pengurangan nilai maksimum dengan minimum.

baris 10 – 11: untuk melakukan seleksi kondisi apabila nilai maksimum sama dengan nilai minimum, maka nilai H[i][j] adalah nol.

baris 12 – 13: untuk melakukan seleksi kondisi apabila nilai maksimum sama dengan nilai pada array rz[i][j], maka nilai H[i][j] dilakukan perhitungan berdasarkan rumus.
baris 14 – 15: untuk melakukan seleksi kondisi apabila nilai maksimum sama dengan nilai pada array gz[i][j], maka nilai H[i][j] dilakukan perhitungan berdasarkan rumus.
baris 16 – 17: untuk melakukan seleksi kondisi apabila nilai maksimum sama dengan nilai pada array bz[i][j], maka nilai H[i][j] dilakukan perhitungan berdasarkan rumus.
baris 18 – 20: untuk melakukan seleksi kondisi apabila nilai maksimum sama dengan nol, maka array S[i][j] diberikan nilai nol, namun apabila kondisi tidak memenuhi, maka array S[i][j] diberikan nilai perhitungan dari nilai delta dibagi nilai maksimum dan dibagi 255.
baris 22: melakukan perhitungan array V[i][j] yaitu hasil perhitungan maksimum dengan 255.
baris 23: melakukan perhitungan array H[i][j] yaitu hasil perhitungan nilai dari indeks tesebut dibagi dengan 2

5.2.2.2 Perhitungan Fitur Statistik HSV

Perhitungan fitur statistik HSV menggunakan citra HSV hasil konversi pada proses sebelumnya. Fitur yang dikonversi yaitu *mean*, deviasi standar, *skewness* dan *kurtosis* pada masing-masing *channel* HSV. Proses perhitungan fitur statistik HSV dapat dilihat pada Algoritme 3.

Algoritme 3: Perhitungan Fitur Statistik HSV

```
1   def getFeatures(gethsv):
2       h,s,v= gethsv[:,:,0],gethsv[:,:,1],gethsv[:,:,2]
3       for i in range (0,p):
4           for j in range (0,l):
5               mean_h += h[i][j]
6               mean_s += s[i][j]
7               mean_v += v[i][j]
8       mean_h = mean_h / pl
9       mean_s = mean_s / pl
10      mean_v = mean_v / pl
11      for i in range (0,p):
12          for j in range (0,l):
13              sd_h += (h[i][j] - mean_h)**2
14              sd_s += (s[i][j] - mean_s)**2
15              sd_v += (v[i][j] - mean_v)**2
16      sd_h = math.sqrt(sd_h / pl)
17      sd_s = math.sqrt(sd_s / pl)
18      sd_v = math.sqrt(sd_v / pl)
19      for i in range (0,p):
20          for j in range (0,l):
21              skewness_h += (h[i][j] - mean_h)**3
22              skewness_s += (s[i][j] - mean_s)**3
23              skewness_v += (v[i][j] - mean_v)**3
24      skewness_h = skewness_h / (pl * (sd_h)**3)
25      skewness_s = skewness_s / (pl * (sd_s)**3)
26      skewness_v = skewness_v / (pl * (sd_v)**3)
27      for i in range (0,p):
28          for j in range (0,l):
29              kurtosis_h += (h[i][j] - mean_h)**4
30              kurtosis_s += (s[i][j] - mean_s)**4
```

```
31              kurtosis_v += (v[i][j] - mean_v)**4
32      kurtosis_h = kurtosis_h / (pl * (sd_h)**4) - 3
33      kurtosis_s = kurtosis_s / (pl * (sd_s)**4) - 3
34      kurtosis_v = kurtosis_v / (pl * (sd_v)**4) - 3
```

Penjelasan:
baris 2: digunakan untuk memecah citra HSV 3 dimensi menjadi citra 1 dimensi yaitu channel h, channel s, dan channel v.
baris 3 – 10: digunakan untuk mendapatkan fitur mean_h, mean_s, mean_v.
baris 11 – 18: digunakan untuk mendapatkan fitur sd_h, sd_s, sd_v.
baris 19 – 26: digunakan untuk mendapatkan fitur skewness_h, skewness_s, skewness_v.
baris 27 – 34: digunakan untuk mendapatkan fitur kurtosis_h, kurtosis_s, kurtosis_v.

5.2.3 Implementasi Ekstraksi Fitur Tekstur

Ekstraksi fitur tekstur digunakan untuk mendapatkan fitur statistik GLCM berdasarkan matriks CM yang telah dilakukan normalisasi. Masukan pada proses ini yaitu citra *grayscale* yang kemudian dilakukan perhitungan matriks CMnya lalu dilakukan normalisasi CM, setelah mendapatkan matriks CM yang telah dilakukan normalisasi maka selanjutnya dilakukan perhitungan fitur statistik GLCM. Fitur yang dihasilkan pada proses ini yaitu berjumlah 10 fitur. Proses perhitungan fitur statistik GLCM dapat dilihat pada Algoritme 4 sampai 6.

5.2.3.1 Menghitung Matriks Co – Occurrence

Proses ini membutuhkan masukan berupa citra *grayscale*. Kemudian citra *grayscale* tadi digunakan untuk melakukan perhitungan matriks *co-occurrence*. Isi dari matriks *co-occurence* yaitu jumlah nilai piksel dari citra yang bersebelahan berdasarkan arah dari sudut yang digunakan. Proses perhitungan matriks *co-occurrence* dapat dilihat pada Algoritme 4.

Algoritme 4: Menghitung Matriks Co - Occurrence

```
1   def getNilaiCM(i, j, citraGray):
2   if((i<0)or(j<0)or(i>=len(citraGray))or(j>=len(citraGray[0]))):
3           return -1
4       else:
5           return citraGray[i][j]
6
7   def getCMmatriks(citraGray, ukCM, P, L, CMsudut0, CMsudut45,
    CMsudut90, CMsudut135):
8       for i in range (0, P):
9           for j in range (0, P-1):
10              if(getNilaiCM(i, j+1, citraGray)!=-1):
11                  CMsudut0[citraGray[i, j], citraGray[i, j+1]] +=
                    1
12      for i in range (0, P-1):
13          for j in range (0, L-1):
14              if(getNilaiCM(i, j+1, citraGray)!=-1):
15                  CMsudut45[citraGray[i+1, j], citraGray[i, j+1]]
                    += 1
16      for i in range (0, P-1):
17          for j in range (0, L):
18              if(getNilaiCM(i, j+1, citraGray)!=-1):
19                  CMsudut90[citraGray[i+1, j], citraGray[i, j]] +=
                    1
```

```
20      for i in range (0, P-1):
21          for j in range (0, L-1):
22              if(getNilaiCM(i, j+1, citraGray)!=-1):
23  CMsudut135[citraGray[i+1, j+1], citraGray[i, j]] += 1
```

Penjelasan:
baris 1 – 5: digunakan untuk mendapatkan nilai piksel yang sesuai untuk perhitungan matriks CM setiap sudut.
baris 8 – 11: digunakan untuk mendapatkan matriks CM 0 ˚
baris 12 – 15: digunakan untuk mendapatkan matriks CM 45 ˚
baris 16 – 19: digunakan untuk mendapatkan matriks CM 90 ˚
baris 20 – 23: digunakan untuk mendapatkan matriks CM 135 ˚

5.2.3.2 Normalisasi Matriks Co – Occurrence

Normalisasi matriks *co-occurrence* bertujuan untuk mengurangi ketergantungan antar matriks dengan cara menjadikan total matriks CM senilai satu. Proses normalisasi dilakukan dengan cara membagi setiap piksel matriks CM dengan jumlah total seluruh matriks CM. Proses perhitungan normalisasi Matris CM dapat dilihat pada Algoritme 5.

Algoritme 5: Normalisasi Matriks Co - Occurrence

```
1   def normalisasiCMmatriks(CMsudut0, CMsudut45, CMsudut90,
2   CMsudut135, ukCM):
3       sumCMsudut0,sumCMsudut45,sumCMsudut90,sumCMsudut135 =
        0,0,0,0
4       for i in range (ukCM):
5           for j in range (ukCM):
6               sumCMsudut0 += CMsudut0[i][j]
7               sumCMsudut45 += CMsudut45[i][j]
8               sumCMsudut90 += CMsudut90[i][j]
9               sumCMsudut135 += CMsudut135[i][j]
10      for i in range (ukCM):
11          for j in range (ukCM):
12              CMsudut0[i][j] = CMsudut0[i][j] / sumCMsudut0
13              CMsudut45[i][j] = CMsudut45[i][j] / sumCMsudut45
14              CMsudut90[i][j] = CMsudut90[i][j] / sumCMsudut90
15              CMsudut135[i][j] = CMsudut135[i][j] / sumCMsudut135
```

Penjelasan:
baris 3 : digunakan untuk menginialisasi nilai sum pada setiap sudut.
baris 4 – 9: digunakan untuk mendapatkan nilai total CM dengan menambahkan semua nilai pada tiap elemen matriks pada masing-masing sudut.
baris 10 – 15: digunakan untuk membagi setiap elemen matriks CM kemudian dibagi dengan total CM.

5.2.3.3 Menghitung Fitur Statistik GLCM

Perhitungan fitur statistik GLCM dilakukan berdasarkan matriks CM yang telah dilakukan normalisasi. Proses ini akan menghasilkan 10 fitur statistik GLCM. Proses perhitungan fitur statistik GLCM dapat dilihat pada Algoritme 6.

Algoritme 6: Menghitung Fitur Statistik GLCM

```
1   def getFiturStatistikGLM(CMsudut0, CMsudut45, CMsudut90,
    CMsudut135, ukCM):
2       Gdissim0,Gdissim45,Gdissim90,Gdissim135=0,0,0,0
3       Genergy0,Genergy45,Genergy90,Genergy135=0,0,0,0
4       Gentropy0,Gentropy45,Gentropy90,Gentropy135=0,0,0,0
```

```
5       Gcontrast0,Gcontrast45,Gcontrast90,Gcontrast135=0,0,0,0
6       Ghomogenity0,Ghomogenity45,Ghomogenity90,Ghomogenity135=0,0
        ,0,0
7       for i in range (ukCM):
8           for j in range (ukCM):
9               Genergy0 += (CMsudut0[i][j])**2
10              Genergy45 += (CMsudut45[i][j])**2
11              Genergy90 += (CMsudut90[i][j])**2
12              Genergy135 += (CMsudut135[i][j])**2
13              if ((CMsudut0[i][j]) > 0): Gentropy0 +=
                CMsudut0[i][j] * math.log10(CMsudut0[i][j])
14              if ((CMsudut45[i][j]) > 0): Gentropy45 +=
                CMsudut45[i][j] * math.log10(CMsudut45[i][j])
15              if ((CMsudut90[i][j]) > 0): Gentropy90 +=
                CMsudut90[i][j] * math.log10(CMsudut90[i][j])
16              if ((CMsudut135[i][j]) > 0): Gentropy135 +=
                CMsudut135[i][j] * math.log10(CMsudut135[i][j])
17              Gcontrast0 += ((i-j)**2) * CMsudut0[i][j]
18              Gcontrast45 += ((i-j)**2) * CMsudut45[i][j]
19              Gcontrast90 += ((i-j)**2) * CMsudut90[i][j]
20              Gcontrast135 += ((i-j)**2) * CMsudut135[i][j]
21              Gdissim0 += abs(i-j) * CMsudut0[i][j]
22              Gdissim45 += abs(i-j) * CMsudut45[i][j]
23              Gdissim90 += abs(i-j) * CMsudut90[i][j]
24              Gdissim135 += abs(i-j) * CMsudut135[i][j]
25              Ghomogenity0 += CMsudut0[i][j] / (1 + (i - j)**2)
26              Ghomogenity45 += CMsudut45[i][j] / (1 + (i - j)**2)
27              Ghomogenity90 += CMsudut90[i][j] / (1 + (i - j)**2)
28              Ghomogenity135 += CMsudut135[i][j] / (1 + (i - jdef
29  getCorrelationVarXVarYMeanXMeanY(CM, ukCM):
30      for i in range (ukCM):
31          for j in range (ukCM):
32              GmeanX = GmeanX + i * CM[i][j]
33      for i in range (ukCM):
34          for j in range (ukCM):
35              GmeanY = GmeanY + j * CM[i][j]
36      for i in range (ukCM):
37          for j in range (ukCM):
38              GvarX = GvarX + ((i - GmeanX)**2) * CM[i][j]
39      for i in range (ukCM):
40          for j in range (ukCM):
41              GvarY = GvarY + ((j - GmeanY)**2) * CM[i][j]
42      GsdX = math.sqrt(GvarX)
43      GsdY = math.sqrt(GvarY)
44      for i in range (ukCM):
45          for j in range (ukCM):
46              Gcorrelation = Gcorrelation + (((i - GmeanY) * (j -
    GmeanY) * CM[i][j]) / (GsdX * GsdY))
```

Penjelasan:

baris 2-6: digunakan untuk menginisialisasi variabel yang akan menjadi variabel penampung fitur.

baris 9-12: digunakan untuk melakukan perhitungan fitur energy di setiap sudutnya.

baris 13-16: digunakan untuk melakukan perhitungan fitur entropy di setiap sudutnya.

baris 17-20: digunakan untuk melakukan perhitungan fitur contrast di setiap sudutnya.

baris 21-24: digunakan untuk melakukan perhitungan fitur dissimilarity di setiap sudutnya.

baris 25-28: digunakan untuk melakukan perhitungan fitur homogeneity di setiap sudutnya.
baris 29-46: digunakan untuk melakukan perhitungan fitur correlation.

5.2.4 Implementasi Information Gain

Proses ini digunakan untuk melakukan seleksi pada fitur-fitur yang dihasilkan pada ekstraksi warna dan tekstur. Seleksi fitur digunakan untuk menghilangkan fitur-fitur yang tidak relevan dan berlebihan sehingga diharapkan mampu didapatkan fitur-fitur terseleksi yang mampu memberikan solusi yang optimal tanpa mengurangi akurasi klasifikasi. Fitur-fitur yang relevan didapatkan dengan melakukan perhitungan nilai entropy pada setiap kelas. Sebelum dilakukan perhitungan information gain, yang lebih dahulu dilakukan adalah merubah data fitur yang numerik menjadi data kategori. Proses implementasi information gain dari merubah data sampai melakukan perhitungan gain di tampilkan pada Algoritme 7 dan 8.

5.2.4.1 Mengubah Data Numerik Menjadi Kategori

Proses ini digunakan untuk mendapatkan data kategori dengan masukan berupa data numerik hasil seleksi fitur. Pada proses ini terdapat proses-proses seperti menghitung nilai minimum, maksimum, jangkauan, panjang kelas dan interval. Sehingga dihasilkan sebuah matriks berupa rentang nilai batas atas dan batas bawah. Kemudian setiap data numerik dimasukkan ke kategori berdasarkan rentangnya. Proses perubahan data numerik menjadi kategori dapat dilihat pada Algoritme 7.

Algoritme 7: Mengubah data numerik menjadi kategori

```
1   for ftrBaris in range(len(seluruhData[0])):
2       wadah = []
3       satuFitur = []
4       if(ftrBaris < (len(seluruhData[0])-1)):
5           for j in range(0, len(seluruhData)):
6            satuFitur.append(float(seluruhData[j][ftrBaris]))
7           n = len(satuFitur)
8           max = np.max(satuFitur)
9           min = np.min(satuFitur)
10          jangkauan = max-min
11          kelas = round(1 + 3.3 * math.log10(n))
12          interval = (jangkauan/kelas)
13          rentang=[]
14          for k in range(kelas):
15              t=[]
16              t.append(min)
17              t.append(min+interval)
18              rentang.append(t)
19              min = (min+interval)
20          tk = len(rentang)-1
21          for j in range(len(satuFitur)):
22              for k in range(len(rentang)):
23                  if(satuFitur[j] >= rentang[k][0] and
                    satuFitur[j] <= rentang[k][1]):
24                      wadah.append(k)
25                  if(satuFitur[j] > rentang[tk][1]):
26                      wadah.append(tk)
27                      break
```

```
28          tes.append(wadah)
29      if(ftrBaris>len(seluruhData[0])-2):
30          for j in range(1, len(seluruhData)):
31              wadah.append(seluruhData[j][ftrBaris])
32          tes.append(wadah)
33  tes=np.transpose(tes)
34  for i in tes:
35      wr.writerow(i)
```

Penjelasan:
baris 1: melakukan perulangan untuk setiap barisnya
baris 2-3: membuat array bernama wadah dan satuFitur
baris 4: seleksi kondisi apabila nilai ftrBaris < len(seluruhData[0])-1)
baris 5: melakukan perulangan dari baris 0 sampai panjang array seluruhData
baris 6: menambahkan data pada indeks seluruhData[j][ftrBaris] ke array satuFitur yang terlebih dahulu dijadikan float
baris 7: menghitung panjang array satuFitur
baris 8-9: mencari nilai maksimum dan minimum
baris 10: mencari nilai jangkauan
baris 11: mencari jumlah kelas
baris 12: mencari nilai interval
baris 13: membuat array rentang
baris 14-19: menambahkan nilai minimum sebagai batas atas dan nilai minimum+interval sebagai batas atas ke array rentang
baris 20: menghitung ukuran rentang dan dikurang 1
baris 21-27: memasukkan data ke kategori berdasarkan batas atas dan batas bawah pada variabel rentang dan menambahkannya ke array wadah
baris 29: seleksi kondisi apabila nilai ftrBaris > len(seluruhData[0])-2)
baris 30-32: perulangan menambahkan baris terakhir ke dalam array wadah
baris 33: melakukan transpose pada array tes
baris 34-35: perulangan untuk menambahkan array tes ke file csv

5.2.4.2 Menghitung Entropy

Perhitungan entropy dilakukan dengan menghitungan probabilitas kemunculan kelas kemudian dilakukan perhitungan sesuai rumus. Proses perhitungan entropy terdapat pada Algoritme 8.

Algoritme 8: Menghitung Entropy

```
1   entropiKelas = 0
2   countKelas = [0]*size
3   jmlData = len(kelas)
4   for i in range(len(kelas)):
5       countKelas[kelas[i][0]] += 1
6   for i in range(len(countKelas)):
7       entropiKelas += -(countKelas[i]/jmlData) *
         math.log2(countKelas[i]/jmlData)
```

Penjelasan:
baris 1: inisialisasi entropiKelas senilai 0
baris 2: inisialisasi list countKleas berukuran [0]*size
baris 3: menghitung panjang kelas
baris 4-5: perulangan untuk menghitung jumlah setiap kelas
baris 6-7: perulangan untuk menghitung entropiKelas

5.2.4.3 Menghitung Gain Setiap Fitur

Perhitungan nilai gain dilakukan pada setiap fiturnya. Semakin tinggi nilai gain menunjukkan bahwa fitur tersebut semakin relevan untuk digunakan. Proses perhitungan nilai gain pada setiap fitur ditunjukkan pada Algoritme 9.

Algoritme 9: Menghitung Gain Setiap Fitur

```
1  gain = [[]]*size
2  for i in range(len(gain)):
3      gain[i] = [[]]*(len(kategori[0]))
4      for j in range(len(gain[i])):
5          gain[i][j] = [0] *
           (max((np.transpose(kategoris))[j])+1)
6  for i in range(len(kategori)):
7      for j in range(len(kategori[i])):
8          gain[kelas[i][0]][j][kategori[i][j]] += 1
9  entropiFitur = [[]] * len(gain[0])
10 tempEntropiFitur = [[]] * len(gain[0])
11 for i in range(len(entropiFitur)):
12     entropiFitur[i] = [0] * len(gain[0][i])
13     tempEntropiFitur[i]= [0] * len(gain[0][i])
14     for j in range(len(entropiFitur[i])):
15         sumValue = 0
16         for k in range(size):
17             sumValue += gain[k][i][j]
18         for k in range(size):
19             if(sumValue == 0):
20                 entropiFitur[i][j] = 0
21             else:
22                 if((gain[k][i][j]/sumValue) == 0):
23                     entropiFitur[i][j] = 0
24                 else:
25                     entropiFitur[i][j] += -
                       (gain[k][i][j]/sumValue) *
                       math.log2(gain[k][i][j]/sumValue)
26                     tempEntropiFitur[i][j] = sumValue
27 gainFitur = [0] * len(gain[0])
28 for i in range(len(gainFitur)):
29     temp = 0
30     for j in range(len(tempEntropiFitur[i])):
31         temp += tempEntropiFitur[i][j] / jmlData *
           entropiFitur[i][j]
32     gainFitur[i] = entropiKelas - temp
```

Penjelasan:

Baris 1: membuat list gain dengan ukuran [[]]*size

baris 2-5: perulangan mengisi ukuran list dengan nilai maksimum dari kategori.

baris 6-8: perulangan untuk menghitung jumlah pada setiap kategori dan disimpan dalam list gain.

baris 9-10: membuat list entropiFitur dan tempEntropiFitur dengan ukuran [[]] * len(gain[0])

baris 12-25 : menghitung entropi tiap kategori di setiap fitur

baris 26: menginisialisasi tempEntropiFitur senilai sumValue

baris 27: membuat list gainFitur dengan ukuran 0] * len(gain[0])

baris 28-32: perulangan untuk menghitung nilai entropi setiap fitur

5.2.4.4 Mengurutkan Nilai Gain dan Mendapatkan Hasil Seleksi Fitur

Proses mengurutkan nilai gain dilakukan untuk mengambil fitur-fitur terseleksi sejumlah fitur yang diinginkan. Proses mengurutkan nilai gain dan mendapatkan hasil seleksi fitur terdapat pada Algoritme 10.

Algoritme 10: Mengurutkan Nilai Gain dan Mendapatkan Hasil Seleksi Fitur

```
1  urutan = sorted(range(len(gainFitur)),
   key=gainFitur.__getitem__, reverse=True)
2  for i in range(25):
3      print(urutan[i], ",", gainFitur[urutan[i]])
```

Penjelasan:
baris 1: mengurutkan nilai gainFitur dari nilai tertinggi ke terendah.
baris 2-3: mengambil nilai gainFitur yang telah diurutkan sejumlah range

5.2.5 Implementasi K – Nearest Neighbor

KNN bekerja dengan memilih jarak terdekat dari kelas-kelas yang ada untuk menentukan kelas dari data baru. Pada penelitian ini, metode yang digunakan untuk mengukur jarak adalah *Euclidean Distance* dimana sebelum melakukan perhitungan jarak akan dilakukan proses normalisasi terlebih dahulu. Proses KNN dapat dilihat pada Algoritme 11 dan 12.

Algoritme 11: Normalisasi Data

```
1   def normalisasi(training, testing):
2       min_lama_train=min(training.ravel())
3       min_lama_test=min(testing.ravel())
4       max_lama_train=max(training.ravel())
5       max_lama_test=max(testing.ravel())
6       min_lama=min_lama_train
7       if(min_lama>min_lama_test):
8           min_lama = min_lama_test
9       max_lama=max_lama_train
10      if(max_lama<max_lama_test):
11          max_lama = max_lama_test
12      max_baru=1
13      min_baru=0
14      data_train=(((training-min_lama)/(max_lama-
        min_lama))*(max_baru-min_baru))+0
15      data_test=(((testing-min_lama)/(max_lama-
        min_lama))*(max_baru-min_baru))+0
16       return data_train, data_test
17  training= normalisasi(training1, testing1)[0]
18  testing= normalisasi(training1, testing1)[1]
```

Penjelasan:
baris 2-6: digunakan untuk menginisialisasi nilai minimum dan maksimum dari training dan testing berupa array training dan testing yang di ravel.
baris 7-9: digunakan untuk mendapatkan nilai minimum.
baris 10-11: digunakan untuk mendapatkan nilai maksimum.
baris 12: menginisialisasi max_baru senilai 1.
baris 13: menginisialisasi min_baru senilai 0.
baris 14: melakukan normalisasi setiap data training.
baris 15: melakukan normalisasi setiap data testing.
baris 16: mengembalikan nilai data training dan testing.
baris 17-18: melakukan pemangilan method normalisasi.

Setelah mendapatkan data yang telah ternormalisasi, proses selanjutnya adalah melakukan perhitungan karak dari data uji ke setiap data latih dan menentukan label kelas pada data uji yang dapat dilihat pada Algoritme 12.

Algoritme 12: Menghitung Jarak dan Menentukan Kelas

```
1  def knn(test, k):
2      euclidean=0
3      list_euclid = []
4      for i in range(0, len(labels_train)):
5          euclidean = (test-training[i])**2
6          euclidean = np.sqrt(sum(euclidean))
7          list_euclid.append((euclidean, labels_train[i]))
8      sorted_euclid = sorted(list_euclid, key=lambda x:x[0])
9  donat,roti_gandum,roti_tawar,telor,fried_chicken,rendang,timun,
   gubis,tomat,strawberry,pisang_ijo,pisang_kuning,jeruk,nasi_mera
   h,oreo,beng_beng,soba_mie,gerry_salut,biskuat,milo_nuggets,genj
   i_pie=0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
10     for i in range(0, k):
11         if (sorted_euclid[i][1] == 0): donat += 1
12         elif (sorted_euclid[i][1] == 1): roti_gandum += 1
13         elif (sorted_euclid[i][1] == 2): roti_tawar += 1
14         elif (sorted_euclid[i][1] == 3): telor += 1
15         elif (sorted_euclid[i][1] == 4): fried_chicken += 1
16         elif (sorted_euclid[i][1] == 5): rendang += 1
17         elif (sorted_euclid[i][1] == 6): timun += 1
18         elif (sorted_euclid[i][1] == 7): gubis += 1
19         elif (sorted_euclid[i][1] == 8):  tomat+= 1
20         elif (sorted_euclid[i][1] == 9): strawberry += 1
21         elif (sorted_euclid[i][1] == 10): pisang_ijo += 1
22         elif (sorted_euclid[i][1] == 11): pisang_kuning += 1
23         elif (sorted_euclid[i][1] == 12): jeruk += 1
24         elif (sorted_euclid[i][1] == 13): nasi_merah += 1
25         elif (sorted_euclid[i][1] == 14): oreo += 1
26         elif (sorted_euclid[i][1] == 15): beng_beng += 1
27         elif (sorted_euclid[i][1] == 16): soba_mie += 1
28         elif (sorted_euclid[i][1] == 17): gerry_salut += 1
29         elif (sorted_euclid[i][1] == 18): biskuat += 1
30         elif (sorted_euclid[i][1] == 19): milo_nuggets += 1
31         elif  (sorted_euclid[i][1]  ==  20):  genji_pie  +=  1
32 nilai = [donat,mie,strawberry,pisang,nasi]
33     kelas=nilai.index(max(nilai))
34     return kelas
```

Penjelasan:
baris 2: inisialisasi variabel Euclidean senilai 0.
baris 3: membuat array list_euclid.
baris 4-7: perulangan untuk menghitung jarak Euclidean dari setiap data testing ke seluruh data training.
baris 8: mengurutkan jarak Euclidean dari yang terkecil.
baris 9: menginisialisasi setiap variabel makanan senilai 0.
baris 10-31: melakukan perulangan untuk menghitung jumlah kelas sesuai dengan nilai k.
baris 32-34: mencari dan mengembalikan kelas yang memiliki nilai terbesar.

5.2.6 Akurasi

Akurasi digunakan untuk menunjukkan berapa besar ketepatan tingkat pengujian dengan nilai yang telah diprediksikan dan menunjukkan keberhasilan sistem yang telah dibangun. Jumlah klasifikasi benar yaitu ketika data uji dan kelas

sebenarnya menghasilkan hasil prediksi yang benar oleh metode klasifikasi kemudian dihitung jumlah hasil prediksi yang benar tersebut. Sedangkan jumlah data uji merupakan keseluruhan dari jumlah data yang akan diprediksi. Proses perhitungan akurasi dapat dilihat pada Algoritme 12.

Algoritme 12: Akurasi

```
1  def akurasi(k):
2      benar = 0
3      for i in range(0, len(testing)):
4          kelas_awal = labels_test[i]
5          hasil_klasifikasi = knn(testing[i], k)
6          if (int(kelas_awal) == hasil_klasifikasi):
7              benar = benar+1
8      akurasi = benar/len(testing)*100
9      print("akurasi: ", akurasi)
```

Penjelasan:

baris 2: inisialisasi variabel benar senilai 0.

baris 3-7: perulangan untuk melakukan pengecekan apakah kelas awal sama dengan kelas dari hasil klasifikasi, jika sama menambahkan jumlah variabel benar senilai satu.

baris 8: menghitung akurasi berdasarkan rumus.

baris 9: menampilkan hasil akurasi di *console*

BAB 6 PENGUJIAN DAN ANALISIS

Pada bab ini, dijelaskan mengenai pengujian dan analisis terhadap tiga pengujian utama yang dilakukan. Pengujian tersebut meliputi variasi nilai *k* pada KNN, pengaruh kombinasi fitur HSV dan GLCM serta pengaruh *Information Gain* terhadap HSV dan GLCM. Tujuannya adalah untuk mengetahui keakuratan sistem yang telah diimplementasikan.

6.1 Skenario Pengujian dan Analisis

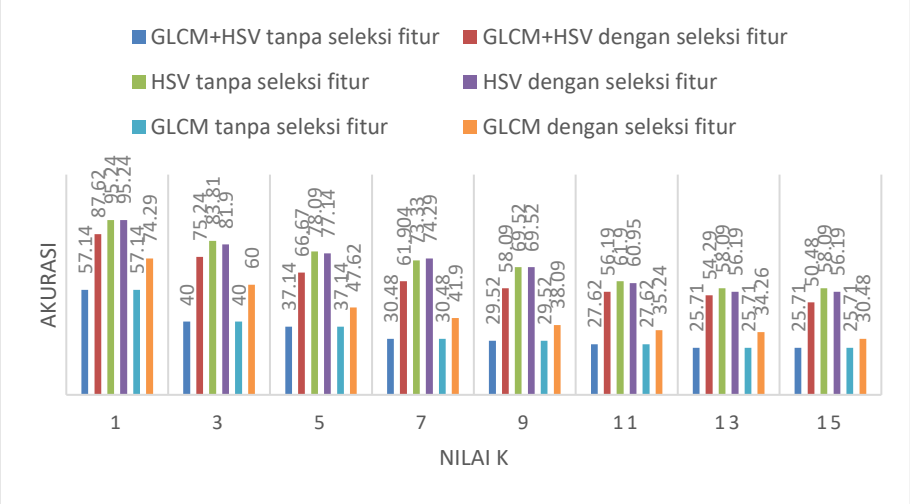
Skenario pengujian dan analisis mengacu pada perancangan yang terdapat pada bab empat. Meliputi pengujian nilai *k*, pengujian pengaruh kombinasi fitur HSV dan GLCM serta pengaruh seleksi fitur *Information Gain*. Setiap pengujian memiliki tujuan yang berbeda-beda. Skenario pertama, yaitu pengujian nilai *k* pada metode KNN yang bertujuan untuk mengetahui nilai *k* optimal yang digunakan untuk melakukan klasifikasi baik pada kombinasi fitur yang digunakan maupun penggunaan seleksi fitur.

Skenario kedua yaitu pengujian menggunakan ekstraksi warna HSV dan tekstur GLCM tanpa seleksi fitur yang bertujuan untuk mengetahui apakah kombinasi kedua ekstraksi fitur ini sudah mencirikan objek dengan baik pada penelitian ini dari setiap citra makanan. Skenario ketiga yaitu pengujian menggunakan HSV dan GLCM dengan menggunakan seleksi fitur, pengujian ini bertujuan untuk mengetahui apakah seleksi fitur dengan menggunakan HSV dan GLCM mampu mendapatkan fitur-fitur terseleksi yang relevan dengan penelitian dan apakah mendapatkan hasil klasifikasi yang lebih baik dibandingan dengan tanpa menggunakan seleksi fitur.

Skenario keempat yaitu pengujian menggunakan HSV tanpa seleksi fitur yang bertujuan untuk mengetahui apakah ekstraksi warna HSV saja cukup digunakan untuk melakukan pemecahan permasalahan pada penelitian ini dan apakah keseluruhan fitur HSV yang digunakan dapat melakukan pencirian objek dengan baik. Skenario kelima yaitu pengujian menggunakan HSV dengan menggunakan seleksi fitur yang bertujuan untuk mengetahui apakah seleksi fitur mampu mendapatkan fitur-fitur relevan pada penelitian dan apakah menghasilkan hasil klasifikasi yang lebih baik dibandingkan dengan tanpa menggunakan seleksi fitur.

Skenario keenam yaitu pengujian menggunakan fitur tekstur GLCM tanpa menggunakan seleksi fitur yang bertujuan untuk mengetahui apakah ekstraksi tekstur GLCM saja cukup digunakan untuk melakukan pemecahan permasalahan pada penelitian ini dan apakah keseluruhan fitur yang digunakan dapat melakukan pencirian objek dengan baik. Skenario ketujuh yaitu pengujian menggunakan GLCM dengan menggunakan seleksi fitur yang bertujuan untuk mengetahui apakah seleksi fitur mampu mendapatkan fitur-fitur relevan pada penelitian dan apakah menghasilkan hasil klasifikasi yang lebih baik dibandingkan dengan tanpa menggunakan seleksi fitur.

6.1.1 Pengujian Nilai K



Gambar 0.1 Pengujian nilai *k*

Pengujian nilai *k* dilakukan pada saat *k* bernilai ganjil. Nilai *k* tersebut terdiri dari satu sampai dengan lima belas. Hasil pengujian nilai *k* dapat dilihat pada Gambar 6.1. Pengujian pertama dilakukan dengan nilai *k*=1 dan mendapatkan akurasi sebesar 57,14% pada kombinasi HSV dan GLCM tanpa seleksi fitur, 87,62% pada kombinasi HSV dan GLCM menggunakan seleksi fitur, 95,24% pada fitur HSV tanpa seleksi fitur dan menggunakan seleksi fitur, 57,14% pada fitur GLCM tanpa seleksi fitur dan 74,26% pada GLCM menggunakan seleksi fitur. Pada pengujian kedua dilakukan dengan nilai *k*=3 dan mendapatkan akurasi sebesar 40% pada kombinasi HSV dan GLCM tanpa seleksi fitur, 75,24% pada kombinasi HSV dan GLCM menggunakan seleksi fitur, 83,81% pada fitur HSV tanpa seleksi fitur, 81,90% menggunakan seleksi fitur, 40% pada fitur GLCM tanpa seleksi fitur dan 60% pada GLCM menggunakan seleksi fitur. Pada pengujian dengan beberapa nilai k selanjutnya terdapat penurunan akurasi setiap kombinasi fiturnya dibandingankan nilai *k*=1 dan k=3 yaitu ketika nilai *k*=5, k=7, *k*=9, *k*=11, *k*=13 dan *k*=15. Hasil akurasi yang didapatkan semakin menurun dengan bertambahnya nilai *k*. Hasil pengujian menunjukkan bahwa semakin kecil nilai *k* semakin optimal dan mendapatkan hasil klasifikasi yang lebih baik baik. Hasil pengujian menyimpulkan bahwa nilai *k* yang paling optimal yaitu ketika *k*=1. Hal ini dikarenakan jika hanya terdapat satu jarak terdekat maka kelas dari data latih langsung diarahkan pada kelas label tanpa terpengaruh ke jarak terdekat yang memiliki fitur mirip dengan kelas lain. Pada Tabel 6.1 merupakan citra yang diklasifikasikan ke kelas lain yang memiliki jarak ketetanggaan terdekat lainnya yaitu ketika *k*=3.

Tabel 0.1 Contoh citra yang salah diklasifikasikan dengan nilai *k*=3

	Jarak terdekat 1	Jarak terdekat 2	Jarak terdekat 3	Kelas sebenarnya	Kelas hasil klasifikasi
Data uji 1	0	20	20	0	20
Data uji 2	1	3	3	1	3
Data uji 3	4	16	16	4	16

Dicontohkan pada data uji pertama dengan hanya menggunakan fitur warna. Kelas yang sebenarnya yaitu 0 (Donat) diklasifikasikan menjadi 20 (Genjie Pie) dimana memiliki kesamaan warna yang di wakilkan pada dua fitur yaitu *mean* H dan deviasi standar H yang dapat dilihat pada Tabel 6.2 untuk data uji sedangkan pada Tabel 6.3 untuk data latih *mean* H pada citra Donat dan Genjie Pie dan Tabel 6.4 untuk data latih deviasi standar H Donat dan Genji Pie. Citra yang mirip tersebut dapat dilihat pada Gambar 6.2.

Donat	Genji Pie
	

Gambar 0.2 Citra yang memiliki kemiripan warna

Tabel 0.2 Nilai *mean* dan deviasi standar pada data uji 1

Data uji	Mean_h	Sd_h
1	6.695404	16.51805

Tabel 0.3 Nilai *mean* H Donat dan Genjie Pie pada data latih

Data latih	Donat	Genjie pie
1	6,885664	6,918844
2	7,007252	8,635248
3	11,41084	7,20176
4	12,22828	3,562576
5	12,02344	3,462612
6	8,453608	7,234512
7	9,869	8,635248
8	7,919368	7,20176
9	7,732964	3,692492
10	7,088916	7,234512

Tabel 0.4 Nilai deviasi standar H Donat dan Genjie Pie pada data latih

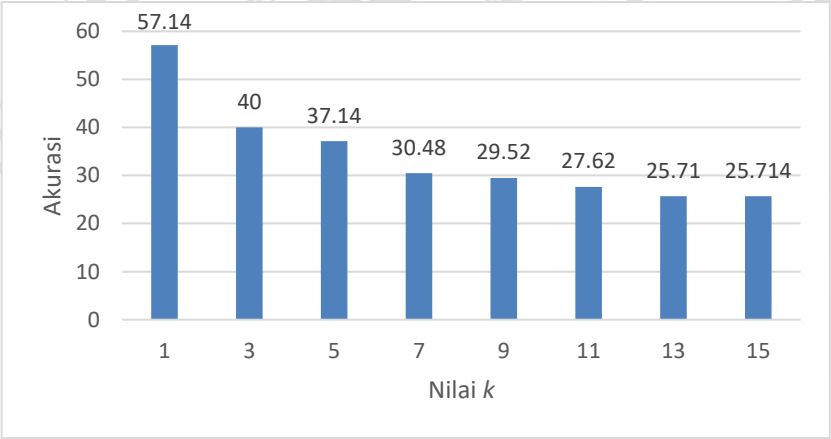
Data latih	Donat	Genjie Pie
1	16,95435	16,93331
2	17,19263	17,24879
3	18,40629	16,23219

Tabel 0.5 Nilai deviasi standar H Donat dan Genjie Pie (lanjutan)

4	17,79947	15,99301
5	17,14957	14,89473
6	18,25729	17,53506
7	16,64408	17,24879
8	18,62901	16,23219
9	18,27501	15,69056
10	17,37755	17,53506

Berdasarkan data yang telah dilakukan analisis, dapat disimpulkan bahwa nilai fitur *mean* H dan deviasi standar H antara citra Donat dan Genji Pie memiliki nilai yang mirip, sehingga apabila dilakukan ketetanggaan terdekat dengan nilai *k* lebih dari satu maka memungkinkan untuk masuk ke kelas yang memiliki warna mirip dengan data uji.

6.1.2 Pengujian Menggunakan HSV dan GLCM Tanpa Seleksi Fitur



Gambar 0.3 Pengujian menggunakan HSV dan GLCM tanpa seleksi fitur

Hasil pengujian menggunakan HSV dan GLCM tanpa seleksi fitur dapat dilihat pada Gambar 6.3. Hasil akurasi terbaik sebesar 57,14% dengan nilai *k*=1. Terdapat banyak data yang salah diklasifikasikan. Hal ini karena terdapat banyak kelas yang memiliki tekstur yang mirip contohnya Strawberry dengan Gerry Salut dan *Milo Nugget*, Jeruk dengan Tomat, Timun, Selada dan Roti Tawar, Biskuat dengan Donat, Jeruk, Tomat dan Telur. Penelitian ini menggunakan fitur GLCM yang cukup banyak, yaitu terdapat empat puluh fitur dengan kelas-kelas yang memiliki banyak kemiripan tekstur. Kemiripan tekstur antar kelas ini menyebabkan fitur-fitur tidak relevan sehingga menyebabkan kesalahan dalam klasifikasi. Selain karena tekstur, juga terdapat kesalahan klasifikasi yang disebabkan dengan kemiripan fitur warna, contohnya pada citra Donat dengan Biskuat, *Fried Chicken* dengan Rendang, *Gerry* Salut dengan Beng-Beng, Rendang dan *Milo Nugget*

No	Kelas Awal	Kelas Klasifikasi
1.		
2.		

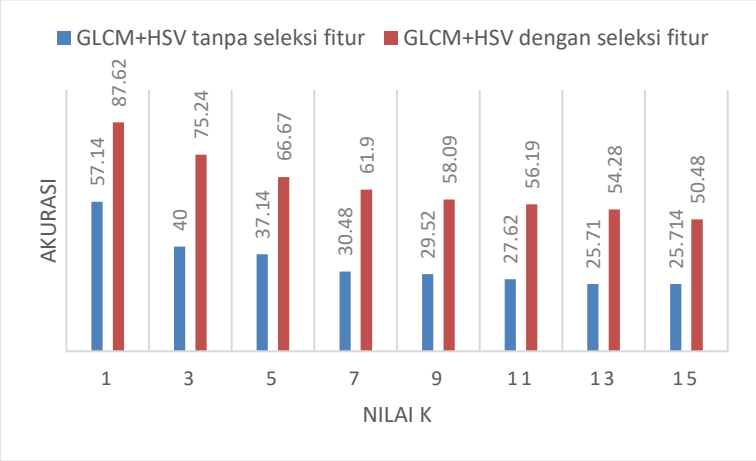
Gambar 0.4 Kelas yang memiliki kesamaan warna dan tekstur

No.	Kelas Awal	Kelas Klasifikasi
1.		
2.		

Gambar 0.5 Kelas yang memiliki hasil segmentasi yang kurang baik

Tidak hanya di tektur dan warna, kesalahan selanjutnya pada citra yang tidak memiliki kemiripan baik tekstur maupun warna, contohnya Telur yang dikenali menjadi Tomat dan juga Pisang Hijau dikarenakan hasil segmentasi Telur yang kurang bagus. Contoh citra yang memiliki kemiripan warna dan tekstur sehingga mengalami kesalahan klasifikasi dapat dilihat pada Gambar 6.4. Pada hasil segmentasi Telur, setiap data latih dan data uji mendapatkan hasil segmentasi yang tidak seragam sehingga tiap datanya dicirikan berbeda-beda dan beberapa data menjadikan putih telur menjadi *background* hitam sehingga menghilangkan informasi warna dan teksturnya. Contoh citra yang memiliki kemiripan warna dan tekstur sehingga mengalami kesalahan klasifikasi dapat dilihat pada Gambar 6.5.

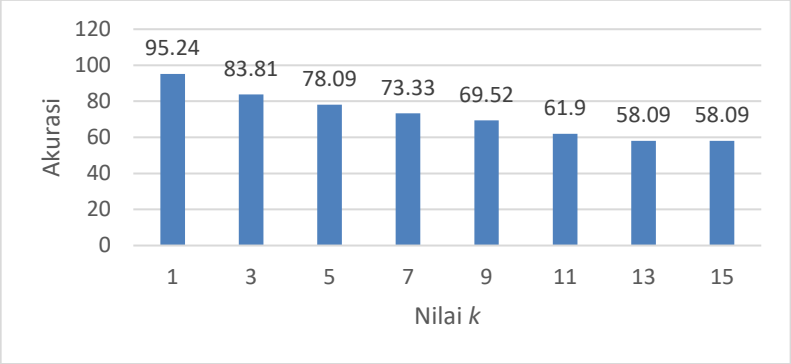
6.1.3 Pengujian Menggunakan HSV dan GLCM dengan Seleksi Fitur



Gambar 0.6 Pengujian menggunakan HSV dan GLCM dengan seleksi fitur

Hasil pengujian menggunakan fitur HSV dan GLCM dengan menggunakan seleksi fitur dapat dilihat pada Gambar 6.6. Pengujian ini mendapatkan hasil akurasi terbaik sebesar 87,62% dengan nilai k=1. Hasil menunjukkan bahwa *Information Gain* mampu mendapatkan fitur-fitur yang relevan berjumlah lima belas dari empat puluh fitur, sehingga mampu meningkatkan akurasi dan mengurangi beban kerja sistem. *Information Gain* mampu meningkatkan akurasi pada pengujian ini hingga 30,48% lebih baik dibanding tangpa menggunakan seleksi fitur.

6.1.4 Pengujian Menggunakan HSV tanpa Seleksi Fitur



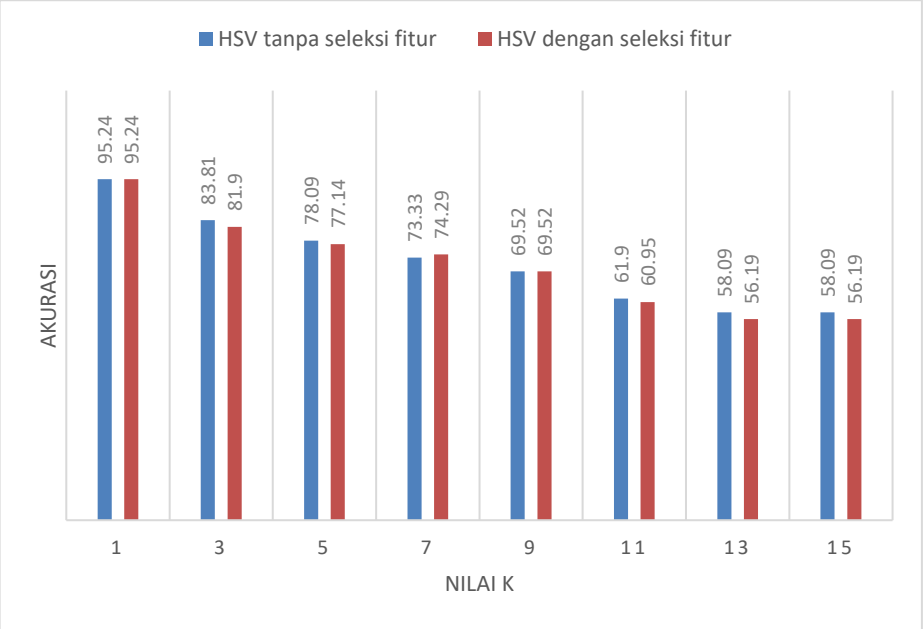
Gambar 0.7 Pengujian menggunakan HSV tanpa seleksi fitur

Hasil pengujian hanya dengan menggunakan fitur *Color Moment* HSV tanpa seleksi fitur dapat dilihat pada Gambar 6.7. Pengujian ini mendapatkan hasil akurasi terbaik sebesar 95,24% dengan nilai *k*=1. Hasil pengujian menunjukkan bahwa *Color Moment* HSV yang berjumlah dua belas fitur pada penelitian ini dapat mencirikan setiap kelas makanan dengan baik sehingga mampu mendapatkan hasil klasifikasi yang baik. Hal ini terjadi karena *Color Moment* HSV dapat melakukan pencirian objek dengan baik meskipun terdapat citra yang memiliki kemiripan warna antar kelas. Mesiki demikian, masih tetap terdapat kesalahan klasifikasi akibat kemiripan warna antar kelas yaitu rendang dengan Biskuat dan Genji Pie, Beng-Beng dengan Biskuat. Contoh citra yang memiliki kemiripan warna tersebut dapat dilihat pada Gambar 6.8.

No.	Kelas awal	Kelas klasifikasi
1.		
2.		

Gambar 0.8 Contoh citra yang memiliki kemiripan warna

6.1.5 Pengujian Menggunakan HSV dengan Seleksi Fitur



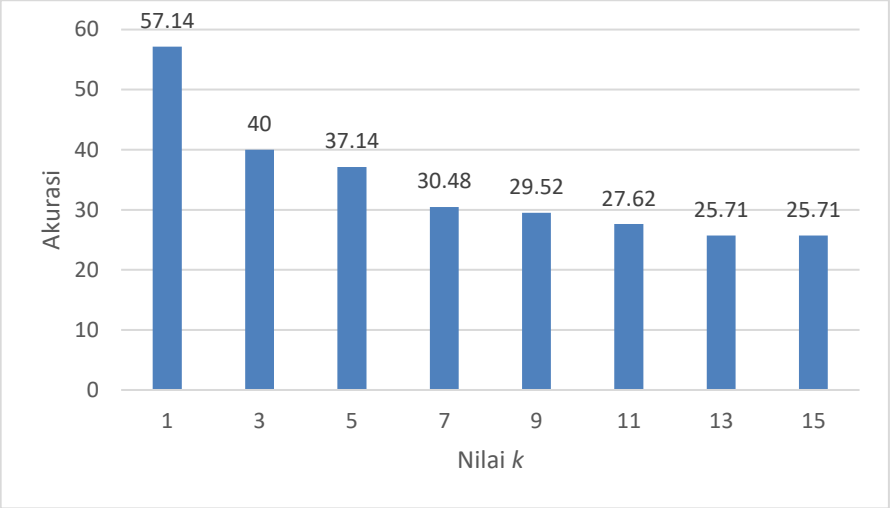
Gambar 0.9 Grafik HSV dengan seleksi fitur

Hasil pengujian hanya dengan menggunakan *Color Moment* HSV dengan menggunakan seleksi fitur dapat dilihat pada Gambar 6.10. Hasil pengujian mendapatkan akurasi terbaik sebesar 95,24% dengan nilai *k*=1. Fitur yang diambil pada seleksi fitur berjumlah delapan dari dua belas fitur HSV. Pengujian ini mendapatkan hasil akurasi yang sama dengan hasil akurasi HSV tanpa menggunakan seleksi fitur pada saat nilai k=1 dan mendapatkan akurasi yang tidak terlalu berjauhan dengan akurasi tanpa menggunakan seleksi fitur pada nilai k yang lain. Terjadi penurunan akurasi pada saat *k*=7. Hal ini terjadi karena kedua belas fitur *Color Moment* dapat mewakili setiap fitur yang diinginkan, apabila empat fitur dihilangkan akan mengurangi informasi warna. Meskipun tidak terdapat peningkatan hasil akurasi yang signifikan, namun mampu mengurangi waktu komputasi dengan menghilangkan empat fitur lainnya. Data yang salah klasifikasi yaitu oreo yang disebabkan hasil segmentasi yang tidak baik, dan beberapa kelas yang memiliki warna bermiripan seperti Rendang, Biskuat, Genji Pie dan Donat. Contoh citra yang memiliki hasil segmentasi tidak baik tersebut dapat dilihat pada Gambar 6.11.

No.	Kelas awal	Kelas klasifikasi
1.		

Gambar 0.10 Contoh citra yang memiliki hasil segmentasi kurang baik

6.1.6 Pengujian Menggunakan GLCM tanpa Seleksi Fitur



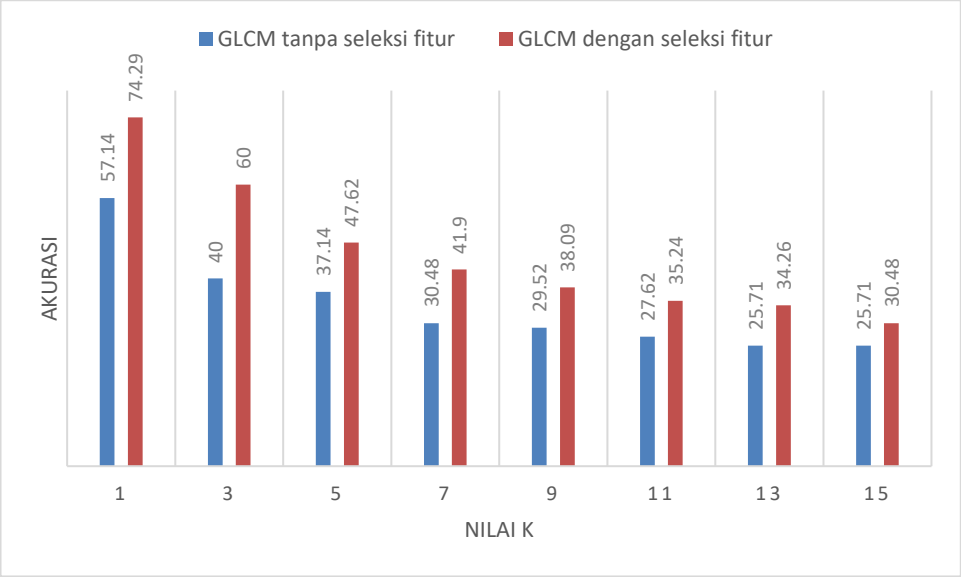
Gambar 0.11 Grafik GLCM tanpa seleksi fitur

Hasil pengujian hanya dengan menggunakan GLCM tanpa seleksi fitur dapat dilihat pada Gambar 6.12. Pengujian ini mendapatkan hasil akurasi terbaik sebesar 57,14% dengan nilai *k*=1. Hasil pengujian menunjukkan bahwa fitur GLCM yang berjumlah empat puluh fitur pada penelitian ini tidak terlalu baik dalam mencirikan tekstur objek. Hal ini terjadi karena terdapat banyak kelas yang memiliki tekstur yang mirip contohnya pada kelas yang salah dilakukan klasifikasi seperti Strawberry dengan Gerry Salut, Jeruk dengan Tomat, Timun, Beng-Beng dengan Millo Nuggets, Fried Chicken, Biskuat dengan Donat, Tomat dengan Pisang Hijau Contoh citra yang salah dilakukan klasifikasi tersebut dapat dilihat pada Gambar. Empat puluh fitur ini terlalu banyak dan setiap fiturnya tidak terlalu mencirikan objek dengan baik karena terdapat kelas yang mirip sehingga menyebabkan kesalahan dalam klasifikasi.

No.	Kelas awal	Kelas klasifikasi
1.		
2.		
		

Gambar 0.12 Citra yang memiliki kemiripan tekstur

6.1.7 Pengujian Menggunakan GLCM dengan Seleksi Fitur



Gambar 0.13 Grafik GLCM dengan Seleksi Fitur

Hasil pengujian hanya dengan menggunakan fitur *GLCM* menggunakan seleksi fitur dapat dilihat pada Gambar 6.7. Pengujian ini mendapatkan hasil akurasi terbaik sebesar 74,28% dengan nilai *k*=1. Hasil pengujian menunjukkan bahwa penggunaan *Information Gain* mendapatkan akurasi yang lebih baik dibanding dengan hasil akurasi fitur GLCM tanpa menggunakan seleksi fitur. Penggunaan seleksi fitur mampu meningkatkan akurasi sebesar 17,14% lebih bail dibandingkan dengan nilai akurasi tanpa menggunakan seleksi fitur.

Fitur yang diambil pada seleksi fitur berjumlah sepuluh fitur dari empat puluh fitur tanpa seleksi fitur. Meski hanya menggunakan sepuluh dari empat puluh fitur yang ada, *Information Gain* mampu melakukan klasifikasi yang lebih baik dibandingkan dengan tanpa menggunakan seleksi fitur. *Information Gain* juga mampu mengurangi waktu komputasi karena fitur yang digunakan tidak terlalu banyak.

BAB 7 PENUTUP

Pada bab ini, dijelaskan kesimpulan berdasarkan pengujian yang telah dilakukan penelitian seleksi fitur *information gain* pada klasifikasi citra makanan menggunakan HSV dan GLCM yang dan saran yang dapat menjadi masukan untuk melakukan penelitian selanjutnya.

7.1 Kesimpulan

Berdasarkan pengujian dan analisis yang telah dilakukan, didapatkan kesimpulan sebagai berikut.

1. Penggunaan kombinasi fitur HSV dan GLCM mendapatkan akurasi yang kurang baik. Hal ini disebabkan kemiripan tekstur antar kelas dan penggunaan fitur GLCM yang terlalu banyak sehingga menyebabkan setiap fiturnya tidak mencirikan objek dengan dan berpengaruh pada hasil klasifikasi. Disamping itu juga terdapat citra yang memiliki kemiripan warna antar kelas. Penggunaan fitur HSV saja mendapatkan hasil klasifikasi yang terbaik dibandingan dua kombinasi fitur lainnya. Dua belas fitur yang digunakan mampu mencirikan objek dengan baik yang dibuktikan dari hasil klasifikasi dan akurasi yang didapatkan. Penggunaan fitur GLCM saja mendapatkan hasil akurasi terendah dibandingkan dengan dua pengujian sebelumnya. Penggunaan fitur GLCM saja yang menggunakan empat puluh fitur menyebabkan setiap fiturnya tidak mencirikan objek dengan baik karena terdapat banyak kelas yang memiliki tekstur mirip serta pengaruh hasil segmentasi yang kurang bagus sehingga membuang informasi penting dari tekstur.
2. Penggunaan seleksi fitur *Information Gain* sangat berperan dalam mendapatkan hasil klasifikasi yang baik dan mampu mengurangi beban kerja sistem dengan mengurangi fitur yang banyak namun mampu mendapatkan hasil yang lebih baik dibandingkan tanpa menggunakan seleksi fitur. Pada pengujian kombinasi fitur HSV dan GLCM dengan mengambil lima belas dari lima puluh dua fitur yang ada, mampu meningkatkan akurasi sebesar 30,48%. Begitu juga ketika menggunakan fitur GLCM saja, seleksi fitur *Information Gain* mampu meningkatkan akurasi sebesar 17,14% dengan mengambil sepuluh fitur dari empat puluh fitur yang digunakan pada penelitian ini. Sedangan Seleksi fitur information gain pada *Color Moment* HSV tidak terlalu mempengaruhi akurasi, dimana akurasi pada seleksi fitur dan tanpa seleksi fitur menghasilkan hasil yang berbeda tipis dikarenakan kedua belas fitur *Color Moment* ini berisi informasi penting mengenai informasi objek, sehingga apabila dilakukan pengurangan fitur akan mengurangi informasi objek tersebut. Meski demikian, Information Gain tetap dikatakan baik untuk seleksi fitur karena dapat menurunkan beban kerja sistem melalui pengurangan fitur.
3. Hasil akurasi terbaik pada penelitian ini yaitu 95,24% dengan hanya menggunakan fitur HSV baik menggunakan seleksi fitur maupun tanpa

menggunakan seleksi fitur dengan nilai k=1. Pada kombinasi HSV dan GLCM dengan seleksi fitur mampu meningkatkan akurasi dari 57,14% menjadi 87,61. Begitu juga dengan fitur GLCM saja menggunakan seleksi fitur yang dapat meningkatkan akurasi dari 57,14% menjadi 74,28%.

7.2. Saran

Penelitian yang telah dilakukan masih memiliki kekurangan dan masih perlu dilakukan pengembangan. Adapun saran dari penelitian ini untuk penelitian selanjutnya yaitu.

1. Melakukan pengembangan penelitian dengan menggunakan metode ekstraksi tekstur lain yang lebih mampu dalam mencirikan objek yang memiliki kemiripan dengan objek lain.
2. Melakukan pengembangan penelitian dengan menggunakan ekstraksi fitur lain, yaitu fitur bentuk.

DAFTAR PUSTAKA

Aditya, C. S. K., Hani'ah, M., Bintana, R. R. & Susciati, N., 2015. *Batik Classification* using *Neural Network with Gray Level Co-occurence Matrix and Statistical Color Feature Extraction*. *International Conference on Information, Communication Technology and System (ICTS),* I(8), pp. 163-167.

Aini, S. H. A., Sari, Y. A. & Arwan, A., 2018. Seleksi Fitur Information Gain untuk Klasifikasi Penyakit Jantung Menggunakan Kombinasi Metode K-Nearest Neighbor dan Naive Bayes. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* II(9), pp. 2546-2554.

Alhassan, Khader, A., 2014. Color And Texture Fusion-Based Method For Content Based Image Retrieval. S2. Sudan university of science and technology. Tersedia di < http://repository.sustech.edu/handle/123456789/10391?show=full>[diakses 12 November 2018].

Azhagusundari, B dan Thanamani, AS., 2013. *Feature Selection based on Information Gain. International Journal of Innovative Technology and Exploring Engineering (IJITEE), Volume* 2, pp.18-21.

Chormunge, S. & Jena, S., 2016. *Efficient Feature Subset Selection Algorithm for High Dimensional Data. International Journal of Electrical and Computer Engineering (IJECE)* , Volume 6, pp. 1880-1888.

Darwiyanto, E., Binawan, B. P. & Junaedi, D., 2017. Aplikasi GIS Klasifikasi Tingkat Kerawanan Banjir Wilayah Kabupaten Bandung Menggunakan Metode Weighted Product. *Ind. Journal On Computing,* II(1), pp. 59-70.

Firmahsyah & Gatini, T., 2016. Penerapan Metode Content-Bbased Filtering Pada Sistem Rekomendasi Kegiatan Ekstra Kulikuler (Studi Kasus Sekolah ABC). Jurnal Teknik Informatika dan Sistem Informasi, Volume 2, pp. 48-56.

Haralick, R.., Shanmugam, & K. Dinstein, I., 1973. *Textural Features for Image Classification*. IEEE Transactions on Systems, Man, and Cybernetics, SMC-3, 610-621.

Hastuti, M. T., Widodo, A. W. & Dewi, C., 2018. Identifikasi Kondisi Kesehatan Ayam Petelur Berdasarkan Ciri Warna HSV Dan Gray Level Cooccurence Matrix (GLCM) Pada Citra Jengger Dengan Klasifikasi K-Nearest Neighbour. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* 2(3), pp. 1054-1062.

He, Y., Khanna, N., Bushey C. J., & Delp, E. J., 2014. Analysis of Food Images: Features and Classification. *ICIP,* 1(4), pp. 2744-27481.

Hoonlor, A. et al, 2015. *UCap: A Crowdsourcing Application for the Visually Ipaired and Blind Persons on Android Smartphone.* Chiang Mai, Thailand, International Computer Science and Engineering Conference (ICSEC).

Jahan, S., H, S. R. & Quadri, S. A., 2018. Bird's Eye Review on Food Image Classification using Supervised Machine Learning. *International Journal of Latest Techology in Engineering, Managements & Applied Science (IJLTEMAS),* VII(3), pp. 153-159.

Joutou, T. & Yanai, K., 2009. A FOOD IMAGE RECOGNITION SYSTEM WITH MULTIPLE KERNEL LEARNING. s.l., 16th IEEE International Conference on Image Processing (ICIP).

Kadir, A., & Susanto. A., 2013. Teori dan Aplikasi Pengolahan Citra. Yogyakarta: penerbit ANDI.

Kusumanto, R., Pambudi, W. S. & Tompunu, A. N., 2011. Klasifikasi Warna Menggunakan Pengolahan Model Warna HSV. *Jurnal Ilmiah Elite Elektro,* II(2), pp. 83-87.

Madenda, S., 2015. Pengolahan Citra & Video Digital. Jakarta:Penerbit Erlangga.

Mohanaiah, P., Sathyanarayana, P. & GuruKumar, L., 2013. Image Texture Feature Extraction Using GLCM Approach. *International Journal of Scientific and Research Publications* (IJSRP), III(5), pp. 1-5.

Mwadulo, M. W., 2016. A Review on Feature Selection Methods For Classification Tasks. *International Journal of Computer Applications Technology and Research,* VI(6), pp. 395-402.

Pamungkas, Adi.2015.Ekstraksi Ciri Citra., Tersedia di: <https://pemrogramanmatlab.com/pengolahan-citra-digital/ekstraksi-ciri-citra-digital/comment-page-1/> [Diakses 28 Agustus 2018]

Patro, S. K. & Sahu, K. K., 2015. *Normalization: A Preprocessing Stage,* India: ResearchGate.

Prasetyo, E., 2011. Pengolahan Citra Digital dan Aplikasinya. Yogyakarta:CV.ANDI OFSET.

Priambodo, A., Dewi, C. & Triwiratno, A., 2015. Implementasi Metode K-Nearest Neighbour untuk Identifikasi Penyakit Tanaman Jeruk Keprok Berdasarkan Citra Daun. *Doro Journal,* 6(8).

Purwati, S., Salimar., Rahayu, S. 2002. Perencanaan Menu Tekanan Darah Tinggi. Penebar Swadaya. Jakarta.

Puspitasari, R. D., Ratnawati, D. E. & Fauzi, M. A., 2018. Optimasi Susunan Gizi Makanan Bagi Pasien Rawat Jalan Penyakit Jantung Menggunakan Real Coded Genetic Algorithm. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* 2(1), pp. 44-52.

Redjeki, S., 2013. Perbandiangan Algoritma *Backpropagation dan K-Nearest Neighbor (K-NN)* untuk Identifikasi Penyakit. Jakarta, Seminar Nasional Aplikasi Teknologi Informasi, pp. 1-5.

Shaltout, N. A., El-Hefnawi, M., Rafea, A. & Moustafa, A., 2014. *Information Gain as a Feature Selection Method for the Efficient Classification of Influenza Based on Viral Hosts. London, U.K, WCE.*

Shapiro & Stockman, 2000*, Computer Vision, Chapter* 5 : *Filtering and Enhancing Images, Department of computer Science & Engineering, University of Washington, Seattle, USA.*

Singh, S. M. & Hemachandran, K., 2012. Content-Based Image Retrieval using Color Moment Gabor Texture Feature. *International Journal of Computer Science,* IX(5), pp. 299-309.

Prasetyo, E. 2012. Data Mining Konsep dan Aplikasi Menggunakan MATLAB. Yogyakarta: ANDI Yogyakarta

Opencv, 2017. Color Conversions. [Online] Doxygen, Tersedia di: https://docs.opencv.org/3.3.0/de/d25/imgproc_color_conversions.htm [Diakses 13 September 2018]