Pengembangan Model Pembelajaran Mesin untuk Klasifikasi Citra Lukisan Menggunakan Self-Organizing Map dengan Library Minisom

Rangga Eka Nanda[1] **Yulius Denny Prabowo**[2]

Informatika, Fakultas Industri Kreatif, Institut Teknologi dan Bisnis Kalbis
Jalan Pulomas Selatan Kav. 22, Jakarta 13210
[1] *Email: rangga.enanda@gmail.com*
[2] *Email: yulius.prabowo@kalbis.ac.id*

Abstract: *This research aims to develop a model to recognize painting types of figurative and non-figurative using self-organizing map (SOM) algorithm. This research used painting images from WikiArt which was formed into figurative and non-figurative types. Methods used in this research implements bag of visual words (BoVW) model to represent image features, SOM algorithm as a classifier, and incremental model as a software development method. Features of an image based on BoVW model formed using scale-invariant feature transform (SIFT) and K-means methods. The BoVW feature representation then classified using SOM which uses rectangular topology and gaussian neighborhood function. The result of this research is an application to recognize painting images with 83.3% accuracy.*
Keywords: *bag of visual word, k-means, painting image, scale-invariant feature transform, self- organizing map*

Abstrak: *Penelitian ini bertujuan untuk membuat model pengenalan citra lukisan figuratif dan non-figuratif, untuk menangani variasi karakteristik seperti komposisi, bentuk dan penggunaan warna digunakan algoritma self-organizing map (SOM) dengan memanfaatkan library Minisom. Penelitian ini menggunakan data citra lukisan figuratif dan non-figuratif dari WikiArt. Penelitian ini menerapkan model bag of visual words (BoVW) untuk merepresentasi fitur citra, algoritma SOM untuk klasifikasi, dan model inkremental sebagai metode pengembangan perangkat lunak. Fitur citra berdasarkan model BoVW dibentuk menggunakan metode ekstraksi fitur scale-invariant feature transform (SIFT) dan K-means. Representasi fitur BoVW kemudian diklasifikasi menggunakan SOM dengan topologi persegi dan fungsi ketetanggaan gaussian. Hasil dari penelitian ini berupa aplikasi pengenalan citra lukisan dengan akurasi sebesar 83.3%.*
Kata kunci: *bag of visual word, citra lukisan, k-means, scale-invariant feature transform, self- organizing map*

I. PENDAHULUAN

Seni lukis telah melalui beberapa perubahan di berbagai era dengan keunikan masing-masing. Perbedaan daerah, pelukis, serta keadaan menghasilkan gerakan seni yang beragam.

Berkembangnya koleksi digital lukisan yang kian bertambah memerlukan adanya otomasi untuk mengidentifikasi, mengarsipkan, serta mengorganisasi data lukisan tersebut. Identifikasi lukisan dengan karakteristik yang kompleks umumnya memerlukan kurator yang ahli pada bidangnya dari segi *style*, *genre*, atau pelukis. Pengelompokan yang sesuai berdasarkan karakteristik yang terdapat pada lukisan secara otomatis berguna untuk membantu pengarsipan digital citra lukisan [1].

Penelitian ini melakukan klasifikasi lukisan jenis figuratif dan non-figuratif, untuk itu diperlukan metode ekstraksi fitur yang sesuai untuk mengidentifikasi subjek pada citra lukisan serta metode klasifikasi untuk mengklasifikasi jenis lukisan. Untuk menangani variasi karakteristik yang terdapat dalam lukisan seperti ragam komposisi, penggunaan bentuk, serta penggunaan warna,

diperlukan metode ekstraksi fitur yang sesuai untuk menggeneralisasi karakteristik tersebut. Visualisasi fitur yang telah diekstraksi juga membantu melihat keterkaitan antara lukisan.

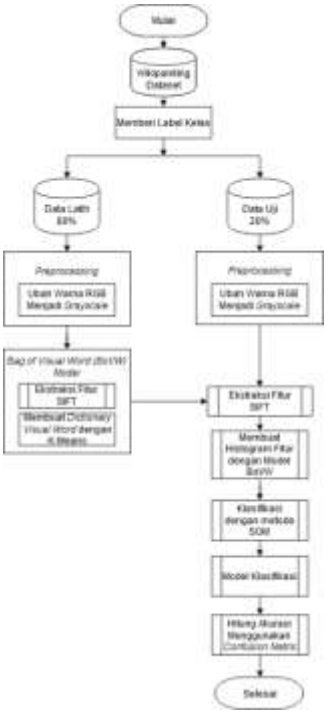
Self-Organizing Map (SOM) digunakan karena selain digunakan untuk pengelompokan data, SOM dapat digunakan untuk mereduksi set fitur pada data input serta memiliki kemampuan untuk memvisualisasikan karakteristik fitur untuk membantu proses analisis [2].

Merujuk pada penelitian [3] [4] [5] yang membahas klasifikasi lukisan, metode *Self-Organizing Map* (SOM) dapat digunakan sebagai *classifier*, karena itu metode SOM digunakan dalam penelitian ini untuk membangun model klasifikasi citra lukisan figuratif dan non-figuratif. Sedangkan untuk representasi fitur digunakan *Bag of Visual Word* (BoVW) berdasarkan [4]. Pembentukan kamus visual menggunakan fitur lokal *Scale-Invariant Feature Transform* (SIFT) dan metode K-means. Dataset yang digunakan adalah subset dari WikiArt [6]. Model yang dihasilkan iimplementasikan dengan aplikasi desktop sehingga pengguna dapat mengoperasikan program melalui aplikasi.

II. METODE PENELITIAN

Penelitian menggunakan metode *Self-Organizing Map* (SOM) berdasarkan representasi fitur model *Bag of Visual Word* (BoVW) menggunakan metode K-means dan fitur SIFT. Dataset WikiArt berupa citra lukisan dengan ekstensi ".jpg" yang dibagi menjadi 27 kelas gerakan seni.

Pada penelitian ini dilakukan klasifikasi lukisan berdasarkan jenis lukisan figuratif dan non-figuratif. Subset data yang digunakan adalah 6 kelas gerakan seni. Gerakan *realism*, *baroque*, dan *impressionism* dikelompokan sebagai figuratif, sedangkan *abstract expressionism*, *action painting*, dan *color field* dikelompokan sebagai non-figuratif. Masing-masing citra sejumlah 50 citra yang dikelompokan menjadi 2 kelas baru. Dataset baru berupa 300 citra lukisan yang dibagi menjadi 150 citra lukisan figuratif dan 150 citra lukisan non-figuratif. 80% (240 citra) dari total data digunakan sebagai data *training* dan 20% (60 citra) digunakan sebagai data *testing*. Kerangka pemikiran penelitian ditunjukan pada Gambar 1.



Gambar 1 Kerangka Pemikiran

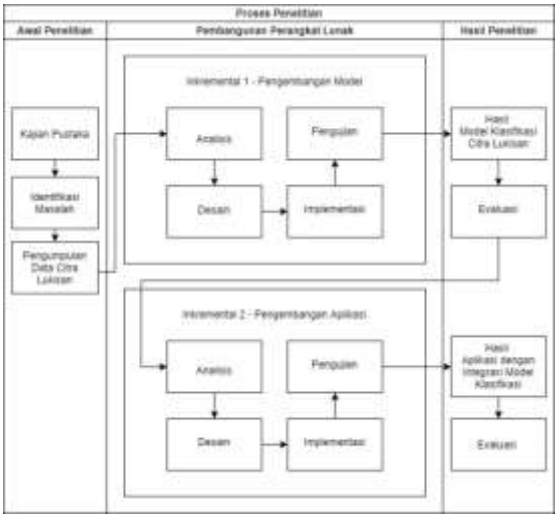
Tahap pertama penelitian diawali dengan melakukan kajian pustaka. Studi literatur mengenai lukisan dilakukan untuk mengetahui karakteristik citra lukisan, representasi fitur *Bag of Visual Word* (BoVW) dari fitur SIFT, serta penggunaan metode SOM untuk melakukan klasifikasi citra.

Metode pembangunan perangkat lunak yang digunakan pada penelitian ini adalah model inkremental. Model inkremental digunakan agar peneliti dapat menambahkan modul fungsi secara bertahap [7]. Model inkremental pada penelitian ini dilakukan melalui dua tahapan, yaitu inkremental satu dan inkremental dua; Pada setiap tahap dilakukan fase analisis, desain, implementasi, dan pengujian.

Pada inkremental satu, model klasifikasi dibangun menggunakan dataset WikiArt. Fase pada inkremental satu yaitu: (1) Analisis: melakukan analisis metode yang digunakan dalam penelitian untuk melakukan klasifikasi

lukisan dengan SOM dan penggunaan representasi fitur model BoVW berdasarkan ekstraksi fitur SIFT; (2) Desain: membuat desain proses pelatihan serta percobaan kombinasi beberapa parameter untuk menentukan arsitektur model terbaik. Uji coba parameter membandingkan jumlah iterasi, jumlah *visual word*, dan jumlah *node* SOM yang ditunjukan pada Tabel 2; (3) Implementasi: Pengkodean dengan beberapa *library* dilakukan untuk membantu proses pembangunan aplikasi. *Preprocess* menerapkan *library* OpenCV untuk mengubah *channel* warna citra RGB menjadi *grayscale*. Pembuatan model BoVW dilakukan dengan menggunakan *library* OpenCV untuk ekstraksi fitur SIFT dan Scikit-Learn untuk implementasi K-means. Pembuatan model klasifikasi *Self-Organizing Map* diimplementasikan menggunakan *library* Minisom [8]. Evaluasi model dengan *confusion matrix* diterapkan menggunakan *library* Scikit-Learn. Visualisasi ditampilkan dengan menggunakan *library* Matplotlib; (4) Pengujian: menguji model berdasarkan hasil *training* dan *testing*. Hasil pelatihan ditunjukan dengan nilai akurasi *training*, *quantization error*, dan *topographic error* terhadap data latih. Hasil *testing* diperoleh melalui pengujian berdasarkan data uji dengan evaluasi *confusion matrix* [9].

Tahap inkremental dua, merupakan pembangunan antarmuka aplikasi berbasis desktop sehingga pengguna dapat melakukan klasifikasi citra lukisan melalui aplikasi. Tahapan pada inkremental dua yaitu: (1) Analisis: melakukan analisis pembuatan aplikasi berbasis desktop dan tujuan pembuatan aplikasi berbasis desktop dengan PyQt5; (2) Desain: membuat desain tampilan antarmuka aplikasi dengan mengintegrasikan model dari inkremen sebelumnya; (3) Implementasi: merealisasikan desain ke dalam kode program. Pembuatan tampilan dilakukan menggunakan aplikasi QtDesigner dari PyQt5; (4) Pengujian: menguji atribut pada *Graphical User Interface* (GUI) untuk memastikan kesesuaian fungsionalitas seperti yang diharapkan untuk melakukan klasifikasi citra lukisan dengan melakukan pengujian *black box* [10]. Proses penelitian ditunjukan pada Gambar 2.



Gambar 2 Proses Penelitian

A. *Bag of Visual Word*

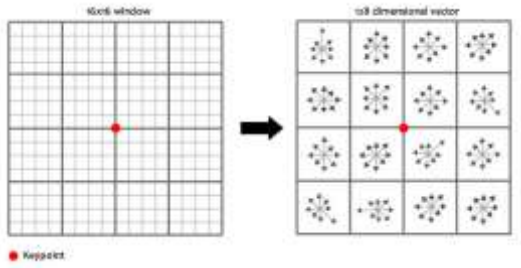
Bag of Visual Word (BoVW) merupakan representasi kumpulan fitur dengan menghitung frekuensi kemunculan fitur visual dalam bentuk histogram dari sebuah citra digital. Model BoVW terinspirasi dari model *Bag of Word* pada *Natural Language Processing* (NLP), di mana model dibentuk dengan menghitung frekuensi kemunculan kata unik dalam setiap dokumen. Pada konteks *computer vision*, kata merupakan fitur visual (*visual word* atau *codeword*), dokumen merupakan citra digital, dan kamus merupakan kamus visual (*visual vocabulary*, *dictionary*, atau *codebook*). Fitur BoVW umumnya digunakan untuk klasifikasi citra, pengenalan objek, dan *image retrieval*. Tahapan untuk membuat fitur BoVW yaitu *sampling*, *coding*, dan *pooling* [11].

Pada penelitian ini, fitur lokal SIFT yang diekstraksi dari kumpulan citra digital digunakan untuk membentuk kamus kata visual. Dikarenakan fitur tidak dapat langsung dihitung layaknya kata pada dokumen, maka fitur yang telah diekstraksi perlu dikelompokan berdasarkan kemiripannya untuk membentuk kumpulan "kata visual" pada kamus visual. Algoritma *clustering* K-means digunakan untuk membentuk kamus kelompok "kata visual".

1. *Scale-Invariant Feature Transform* (SIFT)

Scale-Invariant Feature Transform (SIFT) adalah metode ekstraksi fitur lokal citra dengan menentukan *interest point* atau *keypoint* pada citra. Fitur yang diekstraksi dengan SIFT bersifat invarian terhadap perubahan skala, rotasi, pencahayaan, dan sudut pandang. Fitur yang dihasilkan adalah *descriptor* berupa vektor berukuran 128 untuk setiap *keypoint* yang terdeteksi dalam citra. Fitur SIFT umumnya digunakan untuk melakukan *matching* citra dan pengenalan objek [12].

Tahapan utama dalam metode SIFT yaitu *scale-space extrema detection*, *keypoint localization*, *orientation adaptation assignment*, dan *keypoint descriptor* [12]. Ilustrasi *keypoint* SIFT ditunjukan pada Gambar 3.



Gambar 3 Keypoint Descriptor SIFT [12]

2. K-Means

K-means merupakan algoritma dengan pendekatan *unsupervised* yang digunakan untuk menyelesaikan masalah *clustering*. Algoritma K-means mencoba mengelompokan data ke dalam kelompok atau kluster sejumlah k [13].

Pengelompokan data pada algoritma K-means dilakukan secara iteratif dengan mengukur kedekatan data pada *centroid* sejumlah k yang ditempatkan secara acak. Jarak data dengan *centroid* dihitung dengan jarak Euclidean Iterasi dilakukan hingga tidak terdapat perubahan jarak rerata *centroid* sehingga bentuk kluster konvergen [13]. Rumus penentuan *centroid* dirumuskan sebagai berikut:

$$Centroid = \sum \frac{Ci}{n} \quad (1)$$

Di mana C adalah jarak Euclidean input dengan *centroid* dan n adalah jumlah data dalam sebuah kluster.

B. *Self-Organizing Map*

Self-Organizing Map (SOM) atau Kohonen *network* merupakan jaringan saraf tiruan dengan *layer* tunggal yang menggunakan pendekatan pembelajaran tidak terbimbing (*unsupervised*). SOM memiliki kemampuan mereduksi dimensi sehingga dapat memetakan data dimensi tinggi pada *grid* n-dimensi (umumnya 2-dimensi). Kemampuan reduksi dimensi ini kemudian dapat dimanfaatkan untuk melakukan visualisasi data untuk menganalisis karakteristik data [14], [15].

Visualisasi data pada SOM menerapkan U-matrix (*unified distance matrix*) untuk menggambarkan hubungan data pada *maps* dengan menghitung jarak Euclidean antara node. Topologi SOM yang umumnya digunakan adalah topologi heksagon atau persegi [14], [15].

SOM memiliki konsep pembelajaran kompetitif, di mana vektor (*node*) pada *maps* berkompetisi untuk merepresentasikan data input. SOM menggunakan perhitungan jarak vektor untuk memetakan input pada *maps* fitur 2-dimensi. SOM terdiri dari 3 proses utama yang terdiri dari *competition*, *cooperation*, dan *adaptation* [15].

1. *Competition* adalah tahapan kompetisi representasi setiap *node* terhadap data input. Dikarenakan *node* SOM bersifat *fully connected*, maka jarak kedekatan setiap *node* dibandingkan dengan data input. *Node* dengan nilai yang paling dekat dengan data input disebut sebagai *winner node* atau *Best Matching Unit* (BMU).
2. *Cooperation* adalah tahapan penentuan derajat ketetanggan BMU dengan *node* disekitarnya. BMU akan mengalami pembaruan bobot untuk mengadaptasi data input, sedangkan bobot pada *node* yang berdekatan dengan BMU beradaptasi dengan data input berdasarkan derajat ketetanggaannya.
3. *Adaptation* adalah tahapan pembaruan bobot atau *learning* pada *node* BMU dan

tetangganya. Pembaruan bobot dilakukan sehingga *node* yang representatif dengan input akan cenderung bereaksi kepada input yang memiliki fitur serupa. *Node* dengan nilai serupa cenderung akan membentuk kluster.

Pembelajaran SOM memiliki beberapa tahapan. Tahapan pembelajaran SOM dipaparkan pada Tabel 1.

Tabel 1 Tahapan Pembelajaran Self-Organizing Map

No	*Langkah*
1	*Inisiasi bobot awal pada maps, iterasi (T), learning rate (a), dan nilai threshold ketetanggaan (σ).*
2	*Ulangi langkah 3-8 selama n-iterasi (t).*
3	*Untuk setiap data input x, lakukan langkah 4-6.*
4	*Bandingkan vektor input dengan setiap vektor pada maps; hitung nilai jarak Euclidean vektor input dengan vektor pada indeks i,j.*
5	*Tentukan Best-Matching Unit (BMU) dengan mengambil vektor dengan jarak Euclidean terkecil. Perhitungan Euclidean dirumuskan sebagai berikut:* $$D(i,j) = \sqrt{\sum_{i=1}^{n}(x_t - w_{i,j})^2} \quad (2)$$ *Di mana x adalah data input dan w adalah node maps.*
6	*Untuk BMU dan setiap node vektor yang bertetangga dengan BMU, update bobot vektor pada maps. Perhitungan update dirumuskan sebagai berikut:* $$w_{i,j}(t+1) = w_{i,j}(t) + a(t)h_{ci,j}(t)(x(t) - w_{i,j}(t)) \quad (3)$$ *Di mana w adalah bobot indeks i,j, x adalah data input, dan h merupakan fungsi ketetanggaan.*
7	*Update nilai learning rate sehingga nilainya berkurang seiring proses pelatihan (fungsi decay). Optimasi learning rate dirumuskan sebagai berikut:* $$a(t) = a / (1 + t/(T/2)) \quad (4)$$
8	*Update nilai threshold ketetanggaan sehingga nilai jarak ketetanggaan semakin kecil. Optimasi ketetanggaan dirumuskan sebagai berikut:* $$\sigma(t) = \sigma / (1 + t/(T/2)) \quad (5)$$ *Fungsi ketetanggaan (Gaussian) dirumuskan sebagai berikut:* $$h_{ci,j}(t) = exp\left(-\frac{{d_{ci,j}}^2}{2\sigma^2(t)}\right) \quad (6)$$ *Di mana d adalah jarak Euclidean ketetanggaan dengan BMU.*

1. *Quantization Error*

Quantization Error (QE) merupakan metrik pengukuran kualitas pembelajaran SOM terhadap data input. QE berupa nilai rerata kedekatan sebuah input dengan BMU input tersebut, di mana jarak tersebut menunjukan kesesuaian input dengan *node* SOM [15].

$$QE = \frac{\sum_{t=1}^{T}|x(t) - w_c(t)|}{T} \quad (7)$$

Di mana T adalah jumlah sampel input, x adalah data input, dan w_c adalah bobot vektor BMU.

2. *Topographic Error*

Topographic Error (TE) merupakan metrik pengukuran kualitas proyeksi data input pada *node* SOM. TE berupa persentase BMU pertama dan kedua sebuah input berdampingan (*adjacent*) atau tidak (*non-adjacent*), sehingga nilai yang lebih rendah menunjukan kesesuaian pemetaan sebuah input pada *node* SOM [15].

$$TE = \frac{1}{T}\sum_{t=1}^{T} u(x(t)) \quad (8)$$

Di mana T adalah jumlah sampel data, $u(x(t))$ = 1 jika *non-adjacent*, dan $u(x(t)) = 0$ jika *adjacent*.

C. Konfigurasi Model

Pada proses pelatihan, percobaan kombinasi beberapa parameter dilakukan untuk menentukan arsitektur model klasifikasi terbaik. Uji coba parameter membandingkan jumlah iterasi, jumlah *visual word*, dan jumlah *node* SOM. Konfigurasi model ditunjukan pada Tabel 2.

Tabel 2 Konfigurasi Model

Model	*Visual Word*	*SOM Node*	*Iterasi*
Model_1	*200*	*49*	*1200*
Model_2	*200*	*49*	*2400*
Model_3	*200*	*81*	*1200*
Model_4	*200*	*81*	*2400*

Model_5	*400*	*49*	*1200*
Model_6	*400*	*49*	*2400*
Model_7	*400*	*81*	*1200*
Model_8	*400*	*81*	*2400*

III. HASIL DAN PEMBAHASAN

A. Inkremental Satu

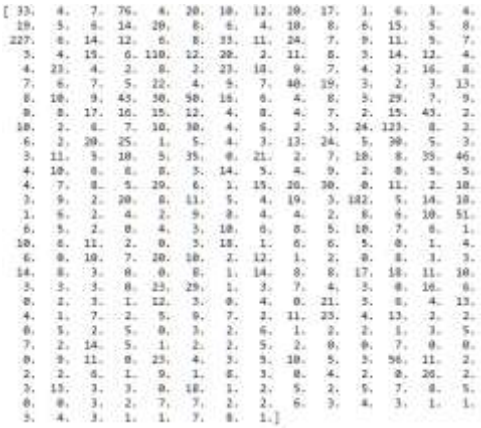
Inkremental satu mencakup pembentukan representasi fitur BoVW serta konfigurasi pembuatan model klasifikasi. Citra input RGB melalui tahap *preprocess* menjadi citra *grayscale*. Proses perubahan channel warna diperlukan karena metode SIFT memerlukan input citra *grayscale*. Perubahan *channel* warna citra diimplementasikan menggunakan *library* OpenCV. Hasil perubahan *channel* warna citra ditunjukan pada Gambar 4.

Gambar 4 Citra Lukisan Grayscale

Keypoint yang terdeteksi ditandai dengan lingkaran berwarna pada citra yang telah diproses. Implementasi ekstraksi fitur SIFT dilakukan menggunakan *library* OpenCV. Hasil deteksi fitur SIFT ditunjukan pada Gambar 5.

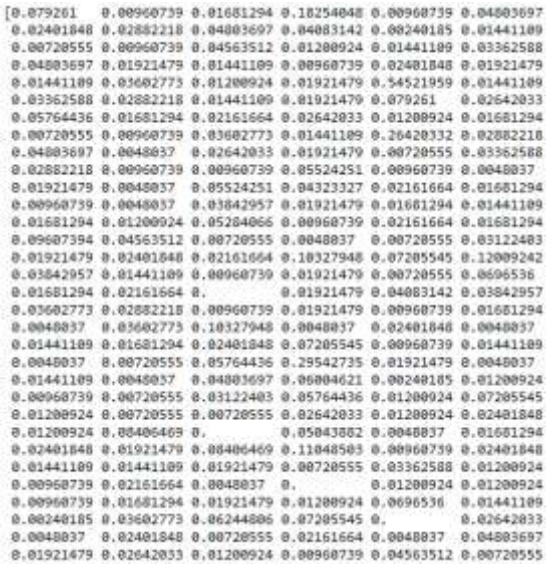
Gambar 5 Citra Lukisan Grayscale dengan Keypoint SIFT

Tahap *preprocess* dan deteksi fitur SIFT dilakukan pada seluruh data *training* untuk membentuk kamus (*dictionary*) representasi fitur BoVW. K-means diterapkan untuk membuat kamus berdasarkan kumpulan fitur keypoint SIFT dari data *training* berdasarkan ukuran kamus yang ditentukan menggunakan *library* Scikit-Learn. Sampel representasi fitur sebuah citra dengan BoVW 400 dimensi ditunjukan pada Gambar 6.



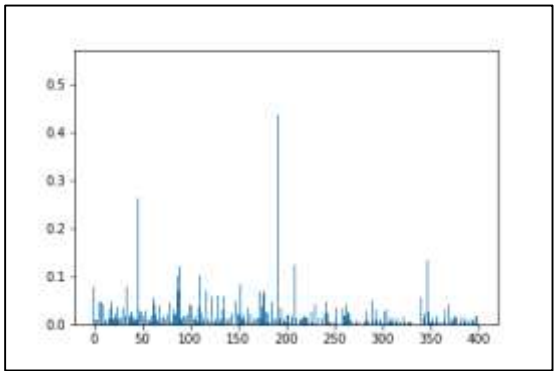
Gambar 6 Representasi Fitur BoVW

Normalisasi dilakukan dengan membagi setiap nilai vektor dengan nilai norm aljabar linear dari keseluruhan vektor fitur tersebut. Implementasi norm aljabar linear dilakukan dengan *library* Numpy. Sampel representasi fitur sebuah citra dengan BoVW 400 dimensi hasil normalisasi ditunjukan pada Gambar 7.



Gambar 7 Representasi Fitur BoVW Hasil Normalisasi

Representasi fitur BoVW kemudian divisualisasikan untuk menunjukan distribusi frekuensi kata visual dengan melakukan *plotting* histogram. *Plotting* diimplementasikan dengan menggunakan *library* Matplotlib. Histogram frekuensi model BoVW sebuah citra ditunjukan pada Gambar 8.



Gambar 8 Histogram Representasi Fitur BoVW

Pelatihan dilakukan berdasarkan konfigurasi pada Tabel 2. Pelatihan dilakukan untuk membentuk bobot SOM serta pemetaan data latih pada node SOM untuk melakukan prediksi menggunakan *library* Minisom. Akurasi *training* setiap model dihitung dengan melakukan prediksi terhadap data latih pada akhir proses pelatihan. Hasil tahapan pelatihan ditunjukan pada Tabel 3.

Tabel 3 Hasil Tahapan Pelatihan Model

Model	*A*	*Akurasi (Training)*	*Recall*	*F1-score*
Model_1	*A1*	*0.854*	*0.420*	*0.067*
Model_2	*A2*	*0.875*	*0.414*	*0.058*
Model_3	*A3*	***0.888***	*0.386*	*0.050*
Model_4	*A4*	*0.867*	*0.381*	*0.054*
Model_5	*A5*	*0.875*	*0.341*	*0.046*
Model_6	*A6*	*0.863*	*0.338*	***0.033***
Model_7	*A7*	*0.875*	*0.309*	***0.033***
Model_8	*A8*	*0.875*	***0.306***	*0.042*

Di mana A merupakan konfigurasi (*visual word_node*_iterasi) dan A1 hingga A8 masing-masing secara berurutan merupakan kombinasi parameter pada Tabel 2.

Akurasi *testing* model didapatkan dengan membandingkan hasil prediksi dengan kelas aktual pada data *testing* yang tidak digunakan untuk membentuk model menggunakan perhitungan *confusion matrix*. Hasil tahapan pengujian ditunjukan pada Tabel 4.

Tabel 4 Hasil Tahapan Pengujian Model

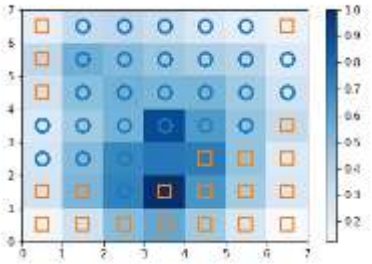
Model	*A*	*Akurasi (Testing)*	*Presisi*	*Recall*	*F1-score*
Model_1	*A1*	*0.733*	*0.74*	*0.73*	*0.73*
Model_2	*A2*	*0.750*	*0.75*	*0.75*	*0.75*
Model_3	*A3*	*0.767*	*0.77*	*0.77*	*0.77*
Model_4	*A4*	*0.700*	*0.73*	*0.70*	*0.69*
Model_5	*A5*	*0.800*	*0.80*	*0.80*	*0.80*
Model_6	*A6*	***0.833***	***0.83***	***0.83***	***0.83***
Model_7	*A7*	*0.767*	*0.78*	*0.77*	*0.76*
Model_8	*A8*	*0.700*	*0.72*	*0.70*	*0.69*

A merupakan konfigurasi (*visual word_node*_iterasi) dan A1 hingga A8 masing-masing secara berurutan merupakan kombinasi parameter pada Tabel 2.

Berdasarkan pengujian, Model_6 menghasilkan nilai akurasi tertinggi sebesar 83.3%. Model_6 menunjukan kemampuan

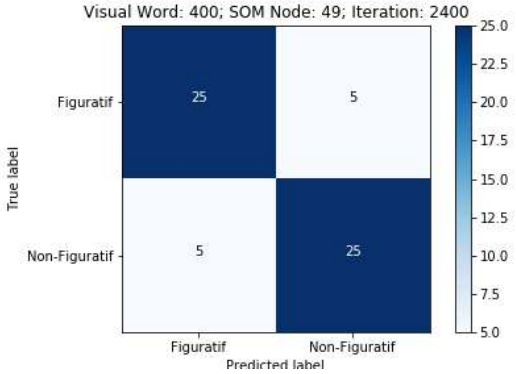
prediktif yang lebih baik dibandingkan model lainnya, sehingga Model_6 dipilih menjadi model akhir.

Visualisasi *node* SOM menampilkan hasil pemetaan data latih pada Model_6. Node dengan notasi lingkaran (o) menunjukan kelas figuratif, sedangkan persegi (□) menunjukan kelas non-figuratif. Intensitas warna menunjukan kedekatan rerata jarak antara satu *node* dengan *node* lainnya. Visualisasi U-matrix Model_6 ditunjukan pada Gambar 9.



Gambar 9 Visualisasi U-matrix Model_6

Confusion matrix menunjukan Model_6 pada kelas positif dan negatif, masing-masing terprediksi tepat sejumlah 25 dan salah sejumlah 5. *Confusion matrix* Model_6 ditunjukan pada Gambar 10



Gambar 10 Hasil Pengujian Confusion Matrix Model_6

B. Inkremental Dua

Inkremental dua mencakup pengembangan tampilan GUI serta implementasi model yang didapat dari tahap inkremental satu. Evaluasi *black box* dilakukan untuk menguji fungsionalitas kelas uji. Berdasarkan hasil pengujian *black box*, aplikasi mampu memberikan keluaran yang sesuai dari fungsi tombol pada kelas uji. Hasil pengujian *black box* ditunjukan pada Tabel 5.

Tabel 5 Hasil Pengujian Black Box

No	*Kelas Uji*	*Harapan*	*Pengamatan*	*Hasil*
1	*Tombol Browse*	*Membuka file dialog; Memuat citra input.*	*Menampilkan file dialog dan memuat citra yang dipilih.*	*Valid*
2	*Tombol Predict*	*Memproses citra input dan menghasilkan kelas prediksi.*	*Menampilkan prediksi berdasarkan citra yang dipilih.*	*Valid*
		Menampilkan visualisasi node SOM dan histogram fitur BoVW.	*Menampilkan visualisasi prediksi pada node SOM dan histogram fitur dari citra yang dipilih.*	*Valid*

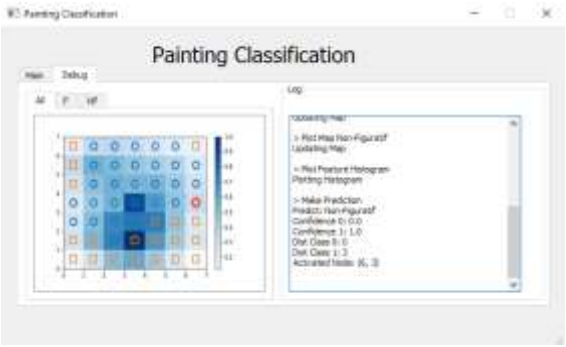
Pembuatan aplikasi desktop dilakukan menggunakan *library* PyQt5. Aplikasi memiliki 2 halaman, "Main" dan "Debug".

Halaman utama "Main" menampilkan judul aplikasi, menu "Main" dan "Debug", kolom gambar citra lukisan dan fitur yang terekstraksi, tombol "Browse" untuk memuat citra lukisan, tombol "Predict" untuk menghasilkan prediksi, serta *text box* dan label untuk menampilkan hasil prediksi. Tampilan antarmuka "Main" ditunjukan pada Gambar 11.

Gambar 11 Antarmuka Aplikasi Halaman "Main"

Halaman "Debug" aplikasi menampilkan pemetaan hasil prediksi *node* pada SOM untuk kedua kelas, kelas figuratif, atau kelas non-figuratif. Selain itu terdapat *text area* untuk

menampilkan aktivitas proses dalam aplikasi. Tampilan antarmuka "Debug" ditunjukan pada Gambar 12.



Gambar 12 Antarmuka Aplikasi Halaman "Debug"

IV. SIMPULAN

Pada penelitian ini, dilakukan klasifikasi citra lukisan figuratif dan non-figuratif menggunakan SOM dan representasi fitur dengan model BoVW. Berdasarkan percobaan yang dilakukan, dapat ditarik simpulan sebagai berikut: (1) Penelitian berhasil menghasilkan model klasifikasi jenis lukisan yang diimplementasikan dengan aplikasi desktop; (2) Model *Bag of Visual Word* (BoVW) dapat merepresentasikan distribusi frekuensi "kata visual" dari citra lukisan; (3) Penggunaan U-matrix dengan SOM masih belum menampilkan perbatasan yang tegas antara kelas yang berbeda sehingga memungkinkan terjadinya kesalahan prediksi; (4) Model_6 dengan konfigurasi "kata visual" 400, 49 *node*, dan epoch 2400 memberikan akurasi *testing* tertinggi sebesar 83.3%.

Selain itu, beberapa saran yang dapat diberikan untuk penelitian selanjutnya yaitu: (1) Mencoba parameter yang berbeda pada pembentukan kamus model BoVW dan ekstraksi fitur SIFT; (2) Mencoba topologi heksagon pada Self-Organizing Map untuk mengkategorikan citra lukisan; (3) Menggunakan metode pembelajaran terbimbing seperti DNN atau SVM.

DAFTAR RUJUKAN

[1] W. R. Tan, C. S. Chan, H. E. Aguirre, and K. Tanaka, "Ceci n'est pas une pipe: A deep convolutional network for fine-art paintings classification," *Proc. - Int. Conf. Image Process. ICIP*, vol. 2016-Augus, pp. 3703–3707, 2016, doi: 10.1109/ICIP.2016.7533051.

[2] L. Yuan, "Implementation of Self-organizing Maps with Python," *ProQuest Diss. Theses*, p. 123, 2018.

[3] S. G. Lee and E. Y. Cha, "Style classification and visualization of art painting's genre using self-organizing maps," *Human-centric Comput. Inf. Sci.*, vol. 6, no. 1, 2016, doi: 10.1186/s13673-016-0063-4.

[4] S. Agarwal, H. Karnick, N. Pant, and U. Patel, "Genre and style based painting classification," *Proc. - 2015 IEEE Winter Conf. Appl. Comput. Vision, WACV 2015*, pp. 588–594, 2015, doi: 10.1109/WACV.2015.84.

[5] A. Lecoutre, B. Negrevergne, and F. Yger, "Recognizing Art Style Automatically in painting with deep learning," *J. Mach. Learn. Res.*, vol. 77, no. 2016, pp. 327–342, 2017.

[6] C. S. Chan, "WikiArt Dataset (Refined)," 2019. [Online]. Available: https://github.com/cs-chan/ArtGAN/tree/master/WikiArt Dataset.

[7] R. S. Pressman, *Software Engineering: A Practitioner's Approach*, 7th ed., vol. 9781118592. McGraw-Hill, 2010.

[8] G. Vettigli, "MiniSom: minimalistic and NumPy-based implementation of the Self Organizing Map." [Online]. Available: https://github.com/JustGlowing/minisom.

[9] P. Cichosz, *DATA MINING ALGORITHMS EXPLAINED USING R*, 1st ed. WILEY, 2015.

[10] S. Roohullah Jan, S. Tauhid Ullah Shah, Z. Ullah Johar, Y. Shah, and F. Khan, "An Innovative Approach to Investigate Various Software Testing Techniques and Strategies," *Int. J. Sci. Res. Sci. Eng. Technol.*, vol. 2, no. 2, pp. 682–689, 2016.

[11] H. Kato and T. Harada, "Image reconstruction from bag-of-visual-words," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 955–962, 2014, doi: 10.1109/CVPR.2014.127.

[12] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, pp. 91–110, 2004.

[13] J. E. Solem, *Programming Computer Vision with Python*, First Edit. O'Reilly Media, 2012.

[14] T. Kohonen, *MATLAB Implementations and Applications of the Self-Organizing Map*. 2014.

[15] C. Kahraman, *Computational Intelligence Systems In Industrial Engineering With Recent Theory And Applications*. Atlantis Press, 2012.