Penerapan *Genetic Neural Network* dalam Pemilihan *Color Palette* untuk Desain Skema Warna

Genetic Neural Network Application in Palette Selection for Scheme Design

Jason Alexander[1], Daniel Ronaldo Pangestu[2], Ferdy Nicolas[3], Lukman Hakim[4]
[1,2,3,4]Program Studi Teknik Informatika, Universitas Bunda Mulia, Jakarta
e-mail: [1]ja669543@gmail.com, [2]ronaldo.pangestu1@gmail.com, [3]ferdy.nicolas@gmail.com, [4]lhakim2710@gmail.com

Abstrak

Jaringan Saraf Tiruan (JST) memiliki kelemahan yang mencolok yaitu diperlukannya parameter yang kompleks untuk dapat memberikan solusi yang tepat terhadap masalah yang diberikan. Untuk mengatasi kelemahan ini, Algoritma Genetika dapat digunakan sebagai metode pemilihan parameter terbaik untuk sebuah JST. Pada penelitian ini, konsep tersebut diterapkan untuk membantu salah satu aspek penting dalam desain yaitu masalah pemilihan warna, dengan menggabungkan kedua teknologi tersebut ke dalam sebuah sistem prediktif. Untuk mencapai tujuan tersebut, penelitian ini membuat model JST berlapis tunggal dengan Algoritma Genetika yang berjalan pada tahap inferensinya. Hasil yang didapat dari penelitian ini adalah bahwa kombinasi JST dan Algoritma Genetika dapat menghasilkan prediksi yang akurat dengan pelatihan yang cukup, variasi data yang memadai, dan perlakuan data yang sesuai; terutama jika menggunakan sumber data yang memiliki nilai sirkular. Apabila salah satu dari ketiga syarat tersebut tidak terpenuhi, maka terdapat risiko terbentuknya zona kritis yang mencegah model untuk dapat menyediakan solusi yang benar.

Kata kunci*— jaringan saraf tiruan, algoritma genetika, palet warna, skema warna*

Abstract

Artificial Neural Networks (ANN) have a distinct flaw, which is the requirement of complex parameters to provide a solution towards a given problem. To mitigate this, a Genetic Algorithm can be used as a method of filtering the best parameter possible for the ANN model. In this research, that concept is applied to aid one of the core aspects of design, which is color selection, by combining the two technologies into a predictive system. To achieve said objective, this research employs a single-layer ANN with a Genetic Algorithm running in its inference engine. The results show that ANN and Genetic Algorithm, when combined, can create an accurate prediction, given enough training time, dataset variation, and correct treatment of its training data; especially when using data that has a circular value. Should any of the three requirements not met, there is a chance that a critical zone will form where the deployed model would not be able to provide the correct solution.

Keywords*— artificial neural networks, genetic algorithms, color palettes, color schemes*

1. PENDAHULUAN

Teknologi komputer memiliki jangkauan penggunaan yang luas. Komputer sebagai salah satu unsur yang bersifat integral terhadap kehidupan masyarakat modern memiliki peran yang penting dalam berbagai bidang dengan proporsi peran yang beragam. Salah satu bidang yang mengintegrasikan perkembangan teknologi dalam bidang aplikatifnya adalah desain. Dalam desain, digunakan berbagai alat bantu untuk menjamin hasil akhir yang memenuhi standar atau mampu menyampaikan konsepsi atau ide pokok yang ingin disampaikan oleh desainer terkait melalui karyanya [1]. Teknologi komputer terbukti dapat menjadi alat bantu yang optimal dalam kegiatan desain karena menjamin presisi dan akurasi desain disertai fasilitas yang memudahkan desainer untuk mengubah, mengembangkan, dan memanipulasi desain ciptaannya [2].

Dari pengantar tersebut, dapat kita simpulkan bahwa keterkaitan antara teknologi dan desain adalah sesuatu yang signifikan, dengan keduanya memiliki hubungan ketergantungan yang bersifat mutual [3], sehingga perkembangan di salah satu bidang akan mempengaruhi bidang lainnya. Sebagai contoh, pada bidang komputer terdapat disiplin ilmu yang memiliki potensi besar untuk diterapkan ke berbagai bidang sebagai alat bantu yang menyediakan layanan inferensi cerdas, yaitu disiplin ilmu Kecerdasan Buatan (*Artificial Intelligence*) yang dengan sendirinya memiliki berbagai cabang multidisiplin berdasarkan penerapannya pada bidang ilmu terkait. Dari sekian banyak cabang yang dihasilkan, diantaranya terdapat dua cabang yang menjadi fokus dari penelitian ini yaitu Algoritma Genetika (*Genetic Algorithm*) dan Jaringan Saraf Tiruan (*Artificial Neural Network*).

Secara terpisah, *Genetic Algorithm* dan *Artificial Neural Network* memiliki bidang implementasi yang berbeda. *Genetic Algorithm* lebih digunakan untuk mencari solusi optimal dan menghasilkan sebuah sistem yang merupakan hasil dari seleksi terhadap masalah terkait disertai skenario hipotetikal yang mungkin terjadi kepadanya [4]. Sedangkan, *Artificial Neural Network* digunakan untuk menciptakan sistem yang memiliki kemampuan untuk melakukan deduksi dan pengambilan keputusan mandiri, atau dapat disebut sebagai sistem yang mampu "berpikir" [5].

Akan tetapi, keduanya dapat digabungkan untuk membentuk sebuah *hybrid* yaitu *Genetic Neural Network* [6]. Gabungan kedua cabang disiplin ilmu *Artificial Intelligence* tersebut kemudian dapat diimplementasikan dalam proses desain, dalam bentuk yang dapat disesuaikan dengan *workflow* dan *work environment* dari desainer yang bersangkutan dan dengan mempertimbangkan berbagai aspek dari sebuah desain yang dapat menjadi *entry point* dari integrasi *Artificial Intelligence* ke dalam konsepsi desain, baik dari segi pembuatan hingga proses *editing* dan *refinement*.

Dalam penelitian ini, aspek desain yang menjadi titik fokus untuk mengimplementasikan kapabilitas dari *Artificial Intelligence* terkait adalah pembuatan skema warna dasar yang dapat diterapkan dalam berbagai jenis desain, mulai dari logo, poster, brosur, dan lain sebagainya. Bidang ini menjadi fokus penelitian karena pemilihan warna merupakan salah satu faktor penting dalam desain yang membantu dalam segmentasi audiens, klasifikasi konten, dan pesan implikatif dari desain yang dibuat, sehingga dapat memiliki signifikansi yang besar terhadap hasil akhir suatu desain yang memanfaatkan daya tarik visual [7].

2. METODE PENELITIAN

2.1 Kajian Pustaka

Penelitian ini akan menghasilkan sebuah aplikasi dengan memanfaatkan *Genetic Neural Network* (GNN) untuk menghasilkan sebuah *color palette* yang dapat menjadi dasar dari sebuah skema warna untuk kegiatan desain.

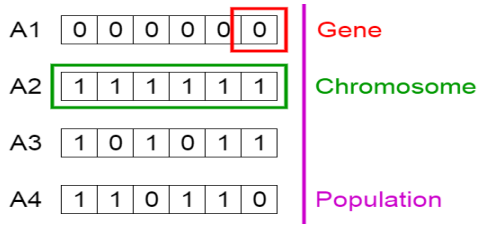
2.1.1 Genetic Algorithm

Genetic Algorithm adalah teknik pencarian yang digunakan dalam komputasi untuk menemukan solusi terbaik untuk masalah *optimization* [8]. *Genetic Algorithm* dikategorikan sebagai heuristik pencarian global dan merupakan hasil pengembangan yang menggunakan teknik yang terinspirasi oleh proses biologi evolusioner seperti warisan, mutasi, seleksi, dan persilangan [4].

Evolusi biasanya dimulai dari populasi individu yang dihasilkan secara acak dan terjadi dalam beberapa generasi. Dalam setiap generasi, nilai *fitness* dari setiap individu dalam populasi dievaluasi, beberapa individu dipilih dari populasi saat ini berdasarkan nilai *fitness* mereka, dan dimodifikasi untuk membentuk populasi baru. Populasi baru digunakan dalam iterasi algoritma berikutnya. Algoritma berakhir ketika jumlah generasi maksimum telah diproduksi, atau tingkat *fitness* yang memuaskan telah dicapai.

Secara sederhana, sebuah *Genetic Algorithm* melakukan seleksi secara terstruktur berdasarkan beberapa tahap sebagai berikut:

1. Inisialisasi
   Algoritma dimulai dengan satu set individu yang disebut **Populasi**. Setiap individu adalah solusi untuk masalah yang ingin diselesaikan. Suatu individu ditandai dengan sekumpulan parameter yang dimilikinya dalam representasi sekumpulan variabel. Masing-masing variabel ini dikenal sebagai **Gen**. Gen digabungkan untuk membentuk **Kromosom**, yaitu parameter lengkap berupa seluruh gen dari individu yang bersangkutan, yang merupakan jawaban yang diberikan individu tersebut terhadap masalah yang diberikan. Ilustrasi antara gen, kromosom, dan populasi dapat digambarkan sebagai berikut [9]:



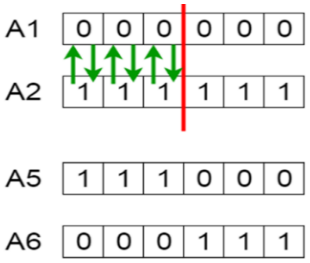
Gambar 1 Ilustrasi hubungan antara gen, kromosom, dan populasi

2. Evaluasi
   Evaluasi menggunakan fungsi *fitness* menentukan seberapa cocok suatu individu terhadap target yang ingin dicapai (fungsi target) dan kemampuan individu tersebut untuk bersaing dengan individu lain [10]. Fungsi ini akan menghasilkan nilai *fitness* untuk setiap individu. Probabilitas bahwa individu tersebut akan dipilih pada tahap seleksi didasarkan pada skor *fitness* ini.

3. Seleksi
   Tujuan fase seleksi adalah untuk memilih individu yang paling cocok dan membiarkan mereka meneruskan gen yang dimiliki ke generasi berikutnya. Dua pasang individu akan dipilih sebagai "orang tua" untuk generasi selanjutnya berdasarkan nilai *fitness* mereka. Individu dengan kebugaran tinggi memiliki kesempatan lebih besar untuk terpilih.

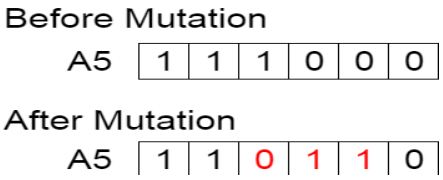
4. *Crossover*
   Crossover adalah fase paling signifikan dalam algoritma genetik. Untuk setiap pasangan orang tua yang akan dikawinkan, titik *crossover* dipilih secara acak dari kromosom, dan seluruh gen sebelum titik tersebut akan disilangkan dengan gen pasangannya sehingga menghasilkan keturunan dengan kombinasi gen baru yang merupakan gabungan dari

kedua gen pasangan individu sebelumnya. Ilustrasi *crossover* dapat dilihat pada gambar berikut [9]:



Gambar 2 Ilustrasi tahap crossover

5. Mutasi
   Pada keturunan baru tertentu yang terbentuk, beberapa gen dapat mengalami mutasi acak dengan probabilitas rendah. Pada setiap titik proses *crossover*, kesalahan dapat terjadi, sehingga mutasi satu titik (*single-point mutation*) cukup umum terjadi [11]. Jenis kesalahan lain yang dapat mempengaruhi lokasi mutasi yang lebih luas seperti delesi, insersi, inversi, dan substitusi juga dapat terjadi. Mutasi terjadi untuk mempertahankan nilai diversitas dalam populasi dan mencegah konvergensi dini. Ilustrasi mutasi dapat dilihat pada gambar berikut [9]:



Gambar 3 Ilustrasi tahap mutasi

6. Terminasi
   Algoritma selesai jika populasi telah mencapai titik konvergen (tidak menghasilkan keturunan yang secara signifikan berbeda dari generasi sebelumnya). Kemudian baru dapat dikatakan bahwa algoritma telah memberikan solusi untuk masalah yang diberikan.

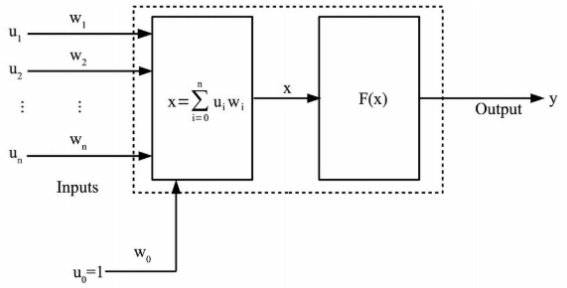
2.1.2 Artificial Neural Network (ANN)

ANN adalah struktur yang dibangun dari elemen dasar yang saling terkoneksi yang disebut neuron. Neuron yang dibuat sebagai basis dari sebuah ANN menyerupai jaringan saraf alami pada otak, yang terdiri dari banyak sel saraf [12]. Deskripsi matematis pertama dari ANN dilakukan oleh McCulloch dan Pitts pada tahun 1943 yang menghasilkan rumus berikut:

$$y = F(\sum_{i=0}^{n}(u_i w_i) + w_0) \quad (1)$$

dimana:
y – nilai output
F – fungsi aktivasi
n – jumlah input
u – nilai input ke-i
w –nilai bobot koneksi ke-i

Selain memiliki definisi matematis, struktur sebuah perceptron juga dapat didefinisikan sebagai sebuah fungsi yang menerima banyak input dan menghasilkan tepat satu output, dengan struktur sebagai berikut [12]:



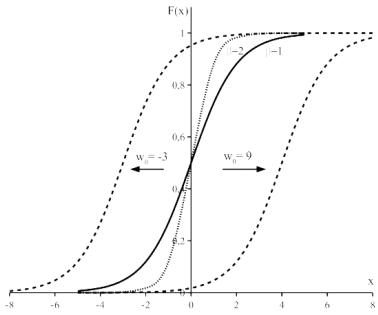
Gambar 4 Struktur sederhana dari sebuah perceptron

Bobot pada koneksi input adalah koefisien yang diatur selama proses pembelajaran. Bobot tersebut merupakan penyimpanan dari pengetahuan yang diperoleh. Setiap nilai input akan dikalikan dengan koefisien bobot dan ditambahkan di akhir kalkulasi. Hasil kalkulasi tersebut merupakan argumen untuk fungsi aktivasi pilihan. Fungsi aktivasi akan menentukan sifat neuron buatan. Hasil dari fungsi aktivasi adalah output dari neuron tersebut. Salah satu fungsi aktivasi yang sering digunakan adalah fungsi aktivasi sigmoid. Rumus fungsi aktivasi sigmoid adalah sebagai berikut:

$$F(x) = \frac{1}{1 + e^{-\beta x}} \quad (2)$$

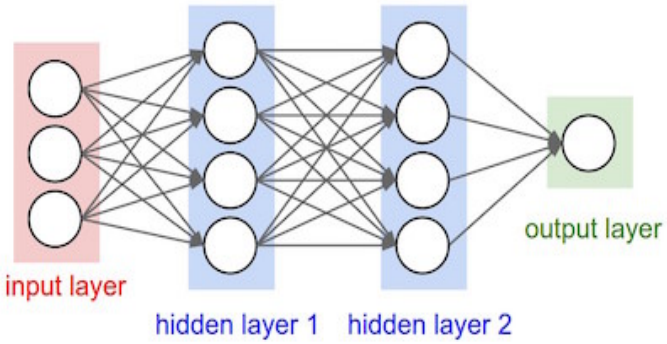
dimana:
F – Tipe fungsi aktivasi
x – Jumlah bobot dikalikan dengan nilai input
β – Parameter kecuraman fungsi sigmoid

Biasanya, fungsi sigmoid yang digunakan adalah fungsi sigmoid unipolar, seperti yang ditunjukkan pada gambar di bawah ini [13]:



Gambar 5 Grafik fungsi sigmoid unipolar

Dalam ANN, neuron disusun ke dalam beberapa lapisan. Masing-masing lapisan terhubung dengan lapisan lain di kedua sisi mereka di mana lapisan di satu sisi terlibat dalam menerima sinyal input yang diperlukan oleh jaringan untuk belajar atau memproses dan lapisan di sisi lain bekerja dengan output dan respons untuk informasi. Di antara dua lapisan ini, ada unit yang tersembunyi yang membentuk lapisan yang disebut *hidden layer*. Susunan perceptron yang berlapis inilah yang menjadi struktur dasar dari *Multi Layer Perceptron* (MLP) [14]. Interkoneksi antara unit dalam *hidden layer* dan unit dalam *output layer* dengan semua unit lain di lapisan sebuah perceptron dikenal sebagai bobot sebuah unit. Bobot menunjukkan kekuatan dependensi/koneksi antar unit. Struktur sederhana dari sebuah MLP dapat dilihat pada ilustrasi berikut:



Gambar 6 Struktur Multi Layer Perceptron

Untuk setiap prosesor dalam lapisan ANN, setiap input dikalikan dengan bobot yang awalnya dikenali, sehingga menghasilkan nilai operasi. Nilai ini selanjutnya diubah oleh nilai ambang batas yang awalnya dihasilkan dan dikirim ke fungsi aktivasi untuk memetakan outputnya. Kemudian output dari fungsi itu dikirim sebagai input untuk lapisan lain, atau sebagai respons akhir dari jaringan jika lapisan adalah yang terakhir. Bobot dan nilai ambang batas biasanya ditingkatkan untuk menghasilkan nilai yang benar dan paling akurat [15].

2.1.3 Genetic Neural Network (GNN)

GNN merupakan algoritma gabungan/*hybrid* dari *Genetic Algorithm* dan *Artificial Neural Network*. Penggabungan algoritma ini pada dasarnya dibagun untuk menutupi kelemahan yang dimiliki oleh algoritma *Artificial Neural Network* seperti *training period* yang lama, parameter yang sangat banyak, dan juga konvergensi lokal yang tidak diinginkan [16]. Penggabungan algoritma ini juga menyebabkan peningkatan akurasi prediksi yang dihasilkan oleh *Artificial Neural Network*, hal ini disebabkan oleh seleksi yang dihasilkan oleh *Genetic Algorithm* merupakan hasil yang telah unggul dari simulasi generasi-generasi sebelumnya.

Dalam GNN, jaringan *perceptron* yang menyusun ANN dikelompokkan sebagai objek komputasi dengan parameter dan nilai *fitness*. Parameter yang dimiliki jaringan tersebut akan dijadikan sebagai gen untuk dioptimalkan oleh *genetic algorithm*. Penggabungan cara kerja ini akan memberikan posisi awal untuk proses *training* yang lebih baik dan memungkinkan lebih sedikit *epoch* yang dibutuhkan dengan akurasi pengujian model yang lebih tinggi [17].

2.1.4 Color Palette & Color Scheme

Color palette atau palet warna adalah sebuah set warna yang terbentuk dari beberapa warna berdasarkan skema warna yang dipilih [18]. Sedangkan *color scheme* atau skema warna adalah suatu *framework* berisikan satu atau lebih palet warna yang dipilih berdasarkan teori warna. [18]. Berdasarkan pengertian sebelumnya dapat diketahui bahwa *color palette* dan *color scheme* merupakan kedua hal yang berbeda namun memiliki ketergantungan satu sama lain untuk membentuk kumpulan warna yang selaras dan seimbang. Keselarasan dan keseimbangan warna ini adalah yang disebut sebagai harmoni warna [19]. Harmoni warna merupakan sesuatu yang sangat penting dalam melakukan teknik desain, hal ini dikarenakan perpaduan warna yang disusun secara kurang tepat dapat menghasilkan kombinasi warna yang dapat menimbulkan kelelahan pada mata [20]. Jenis skema warna dapat diklasifikasikan sebagai berikut:

Gambar 7 Jenis color scheme

1. *Monochromatic*
   Skema warna *monochromatic* terdiri dari variasi satu *hue* dengan menyesuaikan nilai *shading*, *tone*, dan *tint*. Dengan kata lain, skema monokromatik adalah satu warna dengan tingkat konsentrasi warna dasar yang berbeda-beda.
2. *Analogous*
   Skema warna *analogous* terdiri dari kombinasi warna yang bersebelahan pada *color wheel*. Skema warna ini akan menghasilkan kombinasi dari warna yang berbeda, tetapi memiliki dampak visual yang sama.
3. *Complementary*
   Skema warna *complementary* terdiri dari kombinasi warna yang berlawanan pada *color wheel*. Skema warna ini akan menghasilkan kombinasi warna dengan kontras yang tajam, sehingga lebih menarik perhatian.
4. *Triadic*
   Skema warna *triadic* terdiri dari tiga warna yang masing-masingnya memiliki jarak yang sama pada *color wheel*. Skema ini dapat menghasilkan kombinasi warna dengan variasi yang jauh lebih fleksibel jika dibandingkan dengan skema *complementary*, akan tetapi memiliki kontras yang kurang tajam dikarenakan harmoni warna yang lebih terlihat.
5. *Split-Complementary*
   Apabila *complementary* adalah sepasang warna yang berlawanan pada *color wheel*, maka *split complementary* adalah dua pasang warna dengan atribut berlawanan dan memiliki posisi tepat bersebelahan pada color wheel. Skema ini menghasilkan variasi kontras yang sama tajam, dengan pilihan untuk menyesuaikan tingkat ketajaman kontras tersebut.
6. *Tetradic*
   Skema warna *tetradic* terbentuk dari dua pasang skema *complementary*, dengan jarak antar warna yang dibebaskan. Skema ini memiliki tingkat variasi terbesar karena memiliki jumlah kombinasi warna potensial yang paling banyak.

2.2 Rancangan Penelitian

Pada penelitian ini menggunakan metode pengembangan *agile.* Tahap pengembangan pada metode ini dilakukan secara *scrum* yaitu menyelesaikan pekerjaan secara bertahap mengikuti tenggat waktu yang fleksibel [21].

Metode *agile* yang diterapkan pada penelitian ini memiliki enam tahapan utama, sesuai dengan ilustrasi berikut:

Gambar 8 Model pengembangan software agile

1. *Requirements*
   Pada tahap ini dikumpulkan informasi mengenai *software* yang akan dibuat. Informasi yang dikumpulkan akan menjadi basis dari spesifikasi *software* ketika memasuki tahap berikutnya.

2. *Design*
   Tim dibentuk, pendanaan dilakukan, dan lingkungan pengembangan dan persyaratan awal dibahas. Spesifikasi *software* mulai dikembangkan dan disesuaikan.
3. *Development*
   Software mulai dibuat berdasarkan spesifikasi yang ditetapkan pada tahap sebelumnya.
4. *Testing*
   Software diuji dan dites terhadap lingkungan yang terkontrol sebagai simulasi tahap selanjutnya. Apabila *software* sudah melewati standar uji, maka *software* bisa memasuki tahap selanjutnya, yaitu pemakaian secara riil.
5. *Deployment*
   Software dirilis sebagai produk operasional, dengan disertai *maintenance* berkala selama *software* tersebut berada dalam masa efektif operasi.
6. *Review*
   QA (*Quality Assurance*) dilakukan terhadap *software*, dokumentasi dibuat, dan iterasi terhadap versi *software* ditutup.

Dalam penelitian ini, dibuat sebuah *Artificial Neural Network* yang berperan sebagai *inference engine* utama dalam aplikasi yang dikembangkan. Bersamaan dengan itu, dikembangkan pula sebuah *Genetic Algorithm* untuk menentukan bobot yang paling optimal diantara layer-layer yang menyusun ANN. *Artificial Neural Network* yang dibuat memiliki spesifikasi sebagai berikut:

Tabel 1 Spesifikasi Artificial Neural Network yang digunakan

Klasifikasi layer	Jumlah node	Tujuan node	Jangkauan nilai node
Input layer	6	Tema *Analogous*	0 atau 1
		Tema *Triadic*	0 atau 1
		Tema *Monochrome*	0 atau 1
		Nilai *hue*	0 – 1
		Nilai *saturation*	0 – 1
		Nilai *brightness*	0 – 1
Hidden layer	16	Proses utama	0 – 1
Output layer	12	HSB warna terang	0 – 1
		HSB warna netral	0 – 1
		HSB warna gelap (kelompok 1)	0 – 1
		HSB warna gelap (kelompok 2)	0 – 1

Dalam *input layer*, disediakan 6 *node* dengan masing-masing memiliki fungsi yang berbeda. Tiga *node* berfungsi sebagai penentuan pilihan model skema seperti apa yang akan digunakan sehingga hanya bernilai Boolean (0 atau 1) sementara 3 node lainnya masing-masing berfungsi menerima nilai parameter H (*Hue*), S (*Saturation*), dan B (*Brightness*) dari warna dasar yang dipilih.

Setelah input diterima, maka seluruh input akan diproses pada *hidden layer*. Pada *layer* ini, disediakan 16 *node*. Karena ANN yang dibentuk adalah *single-layer* ANN, maka 16 *node* tersebut akan tersusun dalam 1 lapisan saja. Dalam *hidden layer* ini, berlaku sebuah algoritma genetika yang menentukan bobot terbaik dari setiap skenario potensial yang mungkin dipilih ANN.

Setelah selesai melewati *hidden layer*, hasil akhir akan ditampilkan pada *output layer*. Pada *output layer*, terdapat 12 *node* yang terbagi sejumlah 3 *node* untuk setiap kelompok. Pembagian ini berfungsi untuk menampilkan kelompok nilai HSB yang berbeda sebagai kombinasi 5 warna yang membentuk skema akhir.

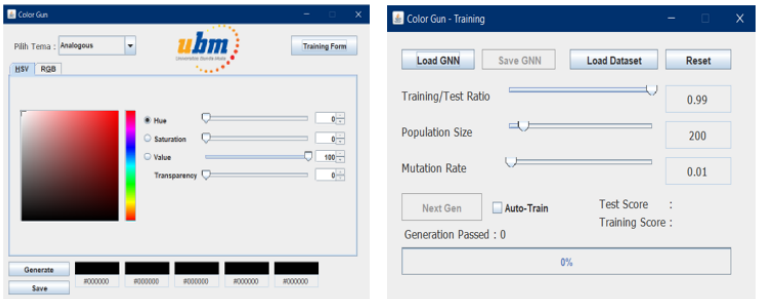
3. HASIL DAN PEMBAHASAN

3.1 Implementasi dan User Interface

Bahasa pemrograman yang peneliti gunakan untuk riset ini adalah Java dengan IDE NetBeans versi 12.1. Dalam penelitian ini, telah disiapkan 3 pilihan skema warna yaitu *analogous*, *triadic*, dan *monochrome*. Terdapat ***jColorChooser*** yang berfungsi untuk memilih warna utama dengan 5 mode pemilihan warna yaitu Swatches, HSV, HSL, RGB, dan CMYK. Akan tetapi, dalam penelitian ini, peneliti hanya menggunakan 2 mode dikarenakan limitasi dataset, yaitu sebagai berikut:

1. HSV
   Model warna HSV (*Hue – Saturation – Value*) mendefinisikan warna dalam terminologi sebenarnya, seperti merah, violet, dan kuning [22]. *Hue* digunakan untuk membedakan warna-warna dan menentukan kadar spektrum dari cahaya. *Saturation* menyatakan tingkat kemurnian suatu warna, yaitu mengindikasikan seberapa banyak warna putih diberikan pada warna. *Value* adalah atribut yang menyatakan banyaknya cahaya yang diterima oleh mata tanpa memperdulikan jenis warna
2. RGB
   RGB (*Red – Green – Blue*) adalah model pewarnaan klasik yang menggabungkan warna merah, hijau, dan biru dengan berbagai proporsi untuk menghasilkan jangkauan warna yang luas. RGB paling baik digunakan dalam teknologi layar, karena mampu menghasilkan warna dengan proporsi yang tepat untuk ditampilkan pada sebuah panel yang memiliki warna dasar putih [23].

Dalam pembuatan aplikasi ini, dibuat 2 buah ***jFrame*** terpisah untuk menampung menu utama dan menu yang digunakan untuk mode pelatihan GNN secara manual apabila *user* ingin melakukan *fine-tuning* secara pribadi untuk menghasilkan GNN baru yang disesuaikan terhadap syarat yang ditetapkan secara personal oleh *user*. Desain *user interface* dari aplikasi yang dibuat adalah sebagai berikut:



Gambar 9 Tampilan main menu & training form

Warna pokok yang dipilih akan diproses menjadi 5 rangkai warna yang disusun dalam *color palette* dengan warna pokok tersebut sebagai acuan utama. Kemudian *user* dapat menyimpan warna dari *color palette* ke dalam file eksternal disertai dengan nilai heksadesimal dari masing-masing warna pada *color palette* tersebut.

Pada *training form*, tombol **Load GNN** berfungsi untuk memuat GNN yang sudah dilatih sebelumnya, sedangkan tombol **Save GNN** digunakan untuk menyimpan hasil *training* yang telah dilakukan. Apabila kita ingin melatih GNN baru, maka dapat digunakan tombol **Load Dataset** untuk menyiapkan file dataset yang sudah disediakan dan tombol **Reset** untuk mengambil kendali penuh atas seluruh parameter sebelum melakukan training, karena rasio *Training/Test* tidak akan dapat diubah apabila pelatihan sudah dimulai atau jika pelatihan dilakukan terhadap GNN yang sudah pernah dilatih sebelumnya.

3.2 Hasil Uji Coba Aplikasi

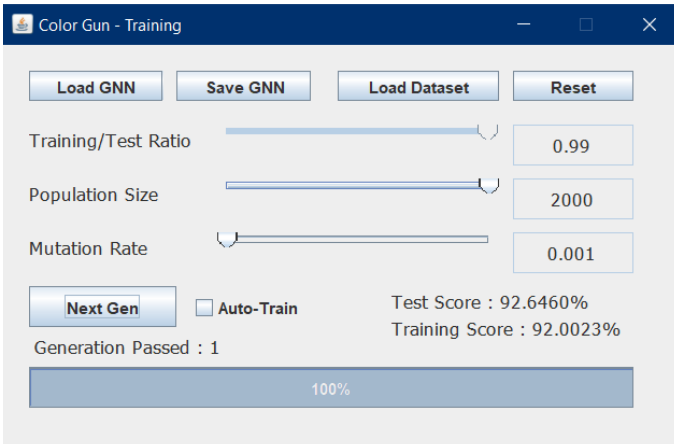
Dataset yang digunakan untuk pelatihan GNN adalah kode heksadesimal dari berbagai kombinasi warna yang merupakan kombinasi yang bersesuaian dengan skema warna tertentu. Dataset ini diperoleh dengan mengekstraksi kode heksadesimal dari *color space* milik SessionsCollege [24] untuk diberi label dan disimpan dalam bentuk file CSV. Setiap entri dataset memiliki atribut berupa 1 label indikator skema, 1 warna dasar, dan 4 warna komplemen. Sebagian dari dataset tersebut dapat dilihat pada gambar 12.

```
a, #AAFFAA, #AAFFD5, #AAF2CE, #D5FFAA, #CEF2AA
t, #AAFFAA, #AAAAFF, #AAAAF2, #FFAAAA, #F2AAAA
m, #AAFFAA, #AAF2AA, #A8E6A8, #A6D9A6, #A3CCA3
a, #FFFFFF, #FFFFFF, #F2F2F2, #FFFFFF, #F2F2F2
t, #FFFFFF, #FFFFFF, #F2F2F2, #FFFFFF, #F2F2F2
m, #FFFFFF, #F2F2F2, #E6E6E6, #D9D9D9, #CCCCCC
a, #E3FFE3, #E3FFF1, #DAF2E6, #F1FFE3, #E6F2DA
t, #E3FFE3, #E3E3FF, #DADAF2, #FFE3E3, #F2DADA
m, #E3FFE3, #DAF2DA, #D1E6D1, #C8D9C8, #BFCCBF
a, #C6FFC6, #C6FFE3, #C2F2DA, #E3FFC6, #DAF2C2
```

Gambar 10 Dataset

Berdasarkan hasil uji coba, didapatkan hasil sebagai berikut:

1. Akurasi optimal yang diperoleh dari hasil *training* terhadap GNN terdapat di kisaran 92%. Sesudah mencapai tingkat ini, proses *training* mengalami tingkat kemajuan yang sangat kecil, dengan perubahan sebesar n x 10^{-4} dengan n adalah nilai random terhadap *test score* dan *training score* yang diperoleh dari hasil pelatihan per generasi. Akurasi yang diperoleh peneliti pada saat pelatihan dihentikan adalah 92.0023% seperti pada gambar 11.



Gambar 11 Hasil training akhir pribadi peneliti

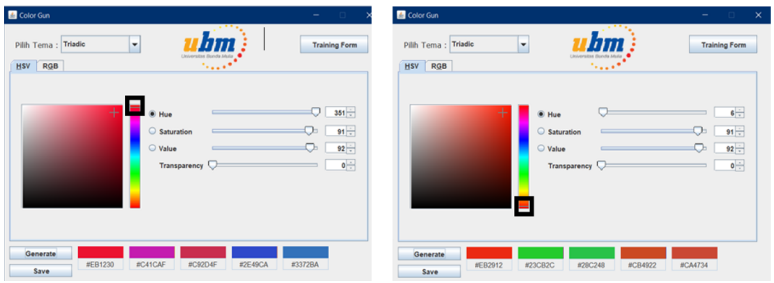
2. Skema warna yang dapat dihasilkan oleh aplikasi terbagi menjadi 3 jenis yang dapat dipilih oleh *user* sesuai dengan kapasitas spesifikasi ANN. Contoh skema warna yang dapat dihasilkan adalah sebagai berikut seperti pada gambar 12 dibawah ini:

Gambar 12 Hasil skema warna dengan berbagai mode skema pilihan.
Kiri: Analogous; **Kanan**: Triadic; **Bawah**: Monochrome

3.3 Kelemahan Algoritma

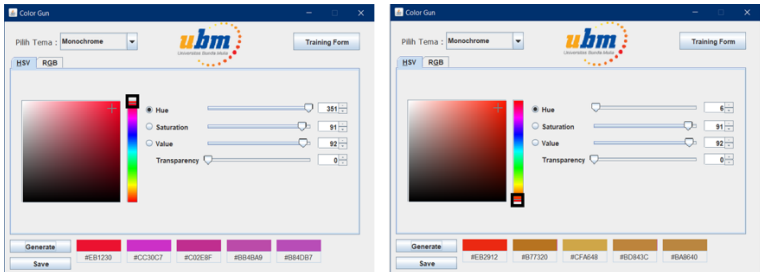
Dalam penelitian ini, algoritma yang diterapkan terhadap model GNN dalam uji coba memiliki kelemahan, yaitu ketidakmampuan model untuk memproses spektrum warna merah. Hal ini disebabkan oleh beberapa faktor, yaitu:

- *Hilangnya sifat sirkular data*
  Dalam menentukan warna, salah satu variabel yang digunakan sebagai parameter adalah *Hue*. *Hue* memiliki nilai yang bersifat sirkular. Akan tetapi, dalam penelitian ini, seluruh data yang dijadikan parameter diperlakukan sebagai data linear, termasuk *Hue* dengan nilai awal 0 dan nilai akhir 359. Hal ini menyebabkan terbentuknya zona kritis dimana hasil yang didapatkan oleh model pada titik temu antara nilai awal dan nilai akhir *Hue* setelah dikonversi paksa menjadi data linear akan selalu bernilai salah. Pada kasus ini, titik temu *Hue* terdapat pada spektrum warna merah dengan titik temu ganda pada perbatasan spektrum warna jingga dan ungu, sehingga warna merah tidak akan pernah mendapatkan hasil skema warna yang benar. Akibatnya, seluruh warna target yang melewati zona kritis tersebut akan mendapatkan hasil yang salah. Contoh replikasi kesalahan tersebut dapat dilihat pada gambar berikut:



Gambar 13 Replikasi kesalahan pada spektrum warna merah mengenai sifat sirkular data.
Kiri: Merah pada spektrum perbatasan ungu; **Kanan**: Merah pada spektrum perbatasan jingga

- *Vanishing gradient problem*
  Vanishing gradient problem adalah masalah yang terjadi pada kondisi dimana nilai yang diperlukan wajib memiliki presisi di sekitar nilai persis 0 atau persis 1, dimana semakin mendekati nilai bulat maka presisi yang didapatkan semakin rendah [25]. Untuk penelitian ini, *vanishing gradient problem* terjadi pada zona kritis yaitu spektrum warna merah, sehingga memaksa model untuk mencari jawaban alternatif terdekat. Replikasi *vanishing gradient problem* pada penelitian ini dapat terlihat pada model spektrum *monochrome*, sebagai berikut:



Gambar 14 Replikasi kesalahan pada spektrum warna merah mengenai vanishing gradient problem. **Kiri**: Merah pada spektrum perbatasan ungu; **Kanan**: Merah pada spektrum perbatasan jingga

4. KESIMPULAN

Berdasarkan hasil penelitian ini, diperoleh kesimpulan bahwa model yang dibuat dengan *training data* dan lama pelatihan sesuai dengan pembahasan dapat menghasilkan GNN dengan akurasi yang cukup baik untuk sebagian besar spektrum warna yang ada untuk model warna HSV dan RGB. Akan tetapi, model tersebut memiliki limitasi pada spektrum warna merah dikarenakan perlakuan data yang kurang tepat sehingga menghasilkan implikasi yang menyebabkan kesalahan dalam sebagian proyeksi hasil akhir. Sebagai referensi, pada penelitian ini digunakan sebanyak 21000 *training data* yang terbagi menjadi 3 model warna dengan lama pelatihan efektif total sekitar 24 jam yang tidak bersifat kontinu. Sangat memungkinkan bahwa akurasi dari model GNN dapat ditingkatkan lebih lagi apabila parameter berupa *training data*, durasi, dan kontinuitas pelatihan ditingkatkan, dengan algoritma yang diperbaiki sehingga lebih efisien.

Dari penjelasan tersebut, peneliti menarik hasil deduksi bahwa keterbatasan dataset, waktu, dan konsistensi dari pelatihan model GNN adalah kelemahan utama dalam penelitian ini, yang berperan besar dalam diperolehnya hasil akhir sesuai temuan dan dokumentasi peneliti. Akan tetapi, peneliti berharap bahwa penelitian ini dapat berfungsi sebagai acuan bagi peneliti lain yang berfokus pada implementasi kecerdasan buatan di bidang desain untuk mempelajari masalah-masalah yang didokumentasikan peneliti selama riset ini berlangsung sebagai petunjuk untuk pengembangan penelitian di bidang ini pada masa mendatang.

5. SARAN

Beberapa saran yang dapat diusulkan peneliti untuk pengembangan penelitian di bidang ini adalah sebagai berikut:

1. Menggunakan metode perlakuan data yang lebih sesuai untuk data sirkular seperti vektorisasi untuk representasi data secara koordinat, karena sistem koordinat dapat digunakan untuk menggambarkan posisi sirkular dengan lebih akurat.
2. Memperluas model warna yang dapat diproses oleh model GNN seperti CMYK, HSL, dan lainnya untuk memenuhi kebutuhan warna dengan akurasi yang berbeda-beda.

3. Mengembangkan metode penyimpanan hasil skema warna yang lebih interaktif dan mudah diaplikasikan langsung dalam kegiatan desain, seperti penyimpanan dalam bentuk gambar, JSON, dan fitur *batch saving* untuk menyimpan banyak skema sekaligus.

DAFTAR PUSTAKA

[1] E. Stolterman and J. Pierce, "Design tools in practice: Studying the designer-tool relationship in interaction design," *Proc. Des. Interact. Syst. Conf. DIS '12*, no. June, pp. 25–28, 2012, doi: 10.1145/2317956.2317961.

[2] J. Li, "Applications of computer-aided design technology in engineering and industry," *Comput. Aided Des. Technol. Types Pract. Appl.*, no. December 2012, pp. 87–102, 2012.

[3] O. Huei, R. Che Rus, and A. Kamis, "Construct Validity and Reliability in Content Knowledge of Design and Technology Subject: A Rasch Measurement Model Approaches for Pilot Study," *Int. J. Acad. Res. Bus. Soc. Sci.*, vol. 10, Mar. 2020, doi: 10.6007/IJARBSS/v10-i3/7066.

[4] D. Setiawan, R. N. Putri, and R. Suryanita, "Implementasi Algoritma Genetika Untuk Prediksi Penyakit Autoimun," *Rabit J. Teknol. dan Sist. Inf. Univrab*, vol. 4, no. 1, pp. 8–16, 2019, doi: 10.36341/rabit.v4i1.595.

[5] A. Pujianto, K. Kusrini, and A. Sunyoto, "Perancangan Sistem Pendukung Keputusan Untuk Prediksi Penerima Beasiswa Menggunakan Metode Neural Network Backpropagation," *J. Teknol. Inf. dan Ilmu Komput.*, vol. 5, no. 2, p. 157, 2018, doi: 10.25126/jtiik.201852631.

[6] R. Mahajan and G. Kaur, "Designing Neural Networks using Genetic Algorithms," *Int. J. Comput. Appl.*, vol. 77, no. 14, pp. 6–11, 2013, doi: 10.5120/13549-1153.

[7] C. R. Lucius and A. Fuad, "Coloring your information: How designers use Theory of Color in creative ways to present infographic," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 277, no. 1, 2017, doi: 10.1088/1757-899X/277/1/012044.

[8] C. Reeves, "Chapter 3 GENETIC ALGORITHMS Part A : Background," *Inf. Sci. (Ny).*, no. May, 1975, doi: 10.1007/978-1-4419-1665-5.

[9] V. Mallawaarachchi, "Introduction to Genetic Algorithms — Including Example Code," 2017. https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3.

[10] S. Mandal, T. A. Anderson, J. S. Turek, J. Gottschlich, S. Zhou, and A. Muzahid, "Learning Fitness Functions for Genetic Algorithms," 2019, [Online]. Available: http://arxiv.org/abs/1908.08783.

[11] S. Marsili Libelli and P. Alba, "Adaptive mutation in genetic algorithms," *Soft Comput.*, vol. 4, no. 2, pp. 76–80, 2000, doi: 10.1007/s005000000042.

[12] L. Pater, "Application of artificial neural networks and genetic algorithms for crude fractional distillation process modeling," no. 2, 2016, [Online]. Available: http://arxiv.org/abs/1605.00097.

[13] S. Tomás, P. Suárez, and S. T. Pérez Suárez, "EasyChair Preprint Design methodology of sigmoid functions for Neural Networks using lookup tables on FPGAs Design methodology of sigmoid functions for Neural Networks using lookup tables on FPGAs," no. October, 2018, doi: 10.13140/RG.2.2.23433.70248.

[14] P. Marius-Constantin, V. E. Balas, L. Perescu-Popescu, and N. Mastorakis, "Multilayer perceptron and neural networks," *WSEAS Trans. Circuits Syst.*, vol. 8, no. 7, pp. 579–588, 2009.

[15] T. Gnanasegaram, "Artificial Neural Network-A Brief Introduction," 2018. https://medium.com/@tharanignanasegaram/artificial-neural-network-a-brief-introduction-572d462666f1.

[16] M. Inthachot, V. Boonjing, and S. Intakosum, "Artificial Neural Network and Genetic Algorithm Hybrid Intelligence for Predicting Thai Stock Price Index Trend," *Comput. Intell. Neurosci.*, vol. 2016, 2016, doi: 10.1155/2016/3045254.

[17] D. Venkatesan, K. Kannan, and R. Saravanan, "A genetic algorithm-based artificial neural network model for the optimization of machining processes," *Neural Comput. Appl.*, vol. 18, no. 2, pp. 135–140, 2009, doi: 10.1007/s00521-007-0166-y.

[18] D. H. Timmons, "Choosing a Color Scheme From the Color Wheel," *The Spruce*, 2019. .

[19] P. Weingerl and D. Javoršek, "Theory of colour harmony and its application," *Teh. Vjesn.*, vol. 25, no. 4, pp. 1243–1248, 2018, doi: 10.17559/TV-20170316092852.

[20] P. Lyons and G. Moretti, "Nine tools for generating harmonious colour schemes," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3101, no. June 2004, pp. 241–251, 2004, doi: 10.1007/978-3-540-27795-8_25.

[21] B. McMillin, *Software Engineering*, vol. 51, no. 2. 2018.

[22] M. Swedia, Ericks Rachmat & Cahyanti, "Algoritma Tranformasi Ruang Warna," *Vis. Bassic6, Vis. Basic.NET dan java*, pp. 1–94, 2010.

[23] P. Menesatti, C. Angelini, F. Pallottino, F. Antonucci, J. Aguzzi, and C. Costa, "RGB color calibration for quantitative image analysis: The '3D Thin-Plate Spline' warping approach," *Sensors (Switzerland)*, vol. 12, no. 6, pp. 7063–7079, 2012, doi: 10.3390/s120607063.

[24] "SessionsCollege," [Online]. Available: https://www.sessions.edu/.

[25] K. Srinivasan, A. K. Cherukuri, D. R. Vincent, A. Garg, and B. Y. Chen, "An efficient implementation of artificial neural networks with K-fold cross-validation for process optimization," *J. Internet Technol.*, vol. 20, no. 4, pp. 1213–1225, 2019, doi: 10.3966/160792642019072004020.