BAB II
TEORI DASAR

2.1 Limbah Tekstil Kain

Limbah kain merupakan potongan sisa kain yang sudah tidak terpakai lagi, tetapi masih bisa digunakan untuk membuat kebutuhan lain dan bisa dimanfaatkan. Limbah kain jenis ini akan menjadi masalah karena tidak diperhatikan lagi keberadaanya dan akan berdampak pada pencemaran lingkungan jika tidak dapat ditangani. Sampah anorganik tidak dapat terurai karena tidak ada aktivitas mikroorganisme pengurai. Oleh karena itu, limbah padat jenis ini harus didaur ulang untuk digunakan kembali. Memanfaatkan limbah kain perca menjadi barang yang dapat digunakan kembali akan memberikan dampak yang sangat baik bagi bumi yaitu mengurangi dampak pemanasan global. Pemanfaatan limbah kain perca sendiri dapat memberikan pemecahan masalah dengan memberikan wawasan kepada masyarakat untuk dapat memanfaatkan limbah kain perca sebagai usaha kecil mereka guna menambah kreatifitas yang bernilai jual sehingga akan dapat membantu perekonomian warga [20]. Berikut merupakan limbah kain yang dapat dilihat pada Gambar 2.1 [21].

Gambar 2. 1 Limbah tekstil kain.

2.2 Machine Learning

Machine Learning (ML) merupakan salah satu cabang dari kecerdasan buatan *Artificial Intelligence* (AI). ML berfokus pada bagaimana membuat komputer atau mesin dapat belajar dari data. Algoritma pembelajaran pada *machine learning* terbagi menjadi beberapa jenis antara lain [9][22]:

1. *Supervised Learning*

   Supervised learning adalah salah satu paradigma dalam *machine learning* di mana algoritma diajarkan untuk mempelajari hubungan antara *input* dan *output* dari data yang telah dilabeli. Dalam *supervised learning*, data pelatihan yang digunakan memiliki *input* dan *output* yang sudah diketahui atau dilabeli. Tujuan utama dari *supervised learning* adalah untuk mempelajari fungsi yang memetakan *input* ke *output*, sehingga ketika dihadapkan pada *input* baru yang belum pernah dilihat sebelumnya, model dapat memprediksi *output* yang sesuai.

2. *Unsupervised Learning*

   Berbeda dengan *supervised learning* yang menggunakan data yang masing-masing telah diberi label, pada algoritma un*supervised learning* data yang digunakan tidak memiliki label. Algoritma ini bermanfaat ketika pola dari data yang diberikan semula tidak diketahui. Algoritma akan belajar untuk menemukan pola yang tersembunyi dari kumpulan data kemudian melakukan pemetaan yang dikenal dengan istilah *clustering*, dimana *dataset* diambil kemudian dikelompokan berdasarkan kemiripannya.

3. *Semi Supervised Learning*

   Semi-supervised learning sebagian data (biasanya dalam proporsi kecil) memiliki label. Sedangkan sebagian besarnya tidak memiliki label. Hal ini biasanya dilakukan ketika pemberian label untuk masing-masing data dirasa terlalu menghabiskan waktu. Misalnya pada kasus deteksi tumor atau penyakit dari data citra hasil CT Scan atau MRI. Akan sangat menyita waktu apabila ahli radiologi memberikan label satu per satu. Algoritma akan memanfaatkan sebagian kecil data yang memiliki label dan memberikan akurasi yang lebih baik dibandingkan *unsupervised learning* secara penuh dimana data tidak diberi label sama sekali.

4. *Reinforcement Learning*

   Reinforcement learning, pembelajaran dilakukan berdasarkan feedback dari lingkungannya. Pembelajaran dilakukan secara berulang. Untuk setiap iterasi (dimana *feedback* diterima) sistem akan mengalami kemajuan.

Teknik ini sangat berguna misalnya untuk melatih robot, melatih mobil tanpa supir (*autonomous vehicle*) dalam menyetir kemudi dan lain-lain. Pada penelitian ini, jenis algoritma *machine learning* untuk sistem klasifikasi digunakan termasuk *supervised learning*, yaitu *K-Nearest Neighbors* (KNN).

2.3 K-Nearest Neighbors

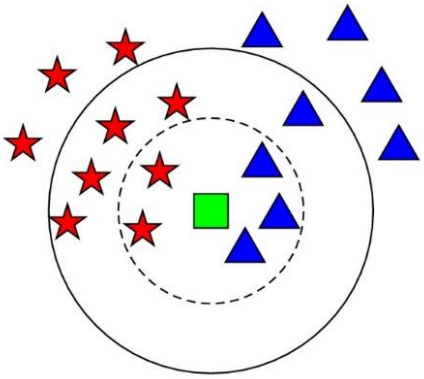
K-Nearest Neighbors termasuk salah satu algoritma *machine learning* yang termasuk kedalam algoritma *supervised learning*. Algoritma ini bekerja dengan cara mengingat data latih (*training set*) kemudian memprediksi label dari suatu obyek baru berdasarkan label-label dari tetangga terdekatnya dalam data latih, tujuan dari algoritma *K-Nearest Neighbors* adalah untuk megklasifikasikan objek baru berdasarkan atribut dan *training samples* [9][23][22].

KNN termasuk ke dalam algoritma *lazy learning* dimana fase *training* hanya bertujuan untuk menyimpan atau mengingat data latih berupa vektor-vektor fitur untuk kemudian dilanjutkan ke fase klasifikasi. Langkah-langkah dalam fase klasifikasi menggunakan algoritma KNN adalah sebagai berikut [24][22]:

1. Menentukan nilai tetangga terdekat K.
2. Menghitung jarak data uji terhadap data-data latih.
3. Mengurutkan data berdasarkan jarak terkecil.
4. Mengklasifikasikan data berdasarkan label mayoritas dari K.

K-Nearest Neighbor (K-NN) memiliki kelebihan yaitu waktu pelatihan KNN singkat, karena hanya perlu menyimpan data pelatihan dalam memori. *K-Nearest Neighbor* juga memiliki kekurangan yaitu membutuhkan nilai k, jarak dari data percobaan tidak jelas dengan tipe jarak yang digunakan, untuk memperoleh hasil yang terbaik, maka harus menggunakan semua atribut atau hanya satu atribut yang telah pasti. Metode *K-Nearest Neighbor* (K-NN) cukup sederhana, tidak ada asumsi mengenai distribusi data dan mudah diaplikasikan. Pemilihan nilai K (jumlah data/tetangga terdekat) ditentukan oleh peneliti. Pemilihan nilai K ini bisa mempengaruhi tingkat akurasi prediksi yang dikerjakan [9][22].

Nilai K pada KNN menentukan berapa banyak tetangga terdekat yang dijadikan acuan untuk menentukan suatu data termasuk ke dalam kelompok yang mana. Jika nilai K adalah 5, maka KNN akan mencari 5 data di dalam *dataset* yang memiliki nilai terdekat dengan titik data yang akan diklasifikasikan. kemudian titik data tersebut akan diklasifikasikan berdasarkan label mayoritas dari 5 data tersebut [24][25]. Untuk lebih jelas dapat diamati pada ilustrasi pada Gambar 2.1 [25][22].



Gambar 2. 2 Ilustrasi KNN dengan K=5.

Gambar 2.2, 5 tetangga terdekat terdiri dari 2 data berbentuk bintang dan 3 segitiga. Sehingga data yang diamati akan diklasifikasikan sebagai anggota bentuk segitiga.

Terdapat beberapa cara untuk menghitung jarak antara titik data yang akan diklasifikasi dengan titik data training. Beberapa diantaranya sebagai berikut [26]:

1. Jarak *Euclidean*

   Jarak *Euclidean* (*Euclidean distance*) adalah perhitungan jarak dari dua buah titik dalam ruang *Euclidean*. Perhitungan jarak ini biasa digunakan untuk membantu proses klasifikasi data seperti pada algoritma *K-Nearest Neighbors*. Jarak *Euclidean* dimensi banyak dapat dihitung mennggunakan rumus:

   $$d(p,q) = \sqrt{(p_1 - q_2)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} \quad (2.1)$$

   Dimana:

   d(p,q) : jarak antara titik p dan q (*distance*: satuan jarak).

   q1,p2 : vektor *Euclidean*, dimulai dari asal ruang (titik awal).

   n : banyaknya dimensi.

2. Jarak Manhattan

Jarak Manhattan adalah jarak antara dua titik yang diukur dengan jumlah selisih pada tiap sumbu dalam koordinat kartesius. Metrik ini terrmasuk norma L1 atau jarak L1. Nama jarak ini berasal dari tata letak jalan di pulau Manhattan yang membentuk segi empat. Jarak Manhattan dapat dihitung dengan menggunakan rumus:

$$d(p,q) = |\, p - q| \tag{2.2}$$

Dimana p dan q vektor:

$$p = (p_1 p_2 p_3, \ldots p_n)\ dan\ q = (q_1, q_2, q_3, \ldots q_n) \tag{2.3}$$

d(p,q) : jarak antara titik p dan q (*distance*: satuan jarak).

q1,p2 : vektor Manhattan, dimulai dari asal ruang (titik awal).

n : banyaknya dimensi.

3. Jarak Chebychev

Jarak Chebychev adalah metrik yang jarak antara dua vektornya adalah selisih maksimum diantara sumbu-sumbunya. Jarak Chebychev dari dua vector $p = (p_1 p_2 p_3, \ldots p_n)\ dan\ q = (q_1, q_2, q_3, \ldots q_n)$ didefinisikan dengan persamaan:

$$D\ (p,q) = maX_i(|p_i - q_1|) \tag{2.4}$$

d(p,q) : jarak antara titik p dan q (*distance*: satuan jarak)

p1,p2 : vector Chebychev (*distance*: satuan jarak)

maxi : selisih maksimum jarak vektor (*distance*: satuan jarak)

Algoritma KNN memiliki keunggulan antara lain [26][25]:

1) Implementasi yang sederhana.
2) Mudah dalam menambahkan data baru.
3) Tahan terhadap data latih yang *noisy*.
4) Memiliki kemampuan untuk memodelkan masalah klasifikasi yang rumit menggunakan kumpulan pendekatan lokal yang lebih sederhana.
5) Merawat informasi yang terdapat dalam data latih.

Selain memiliki keunggulan, algoritma ini juga memiliki beberapa kelemahan, antara lain [9]:

1) Sensitif terhadap data latih yang tidak seimbang (*unbalanced*).
2) Penggunaan memori yang tinggi terutama jika data pelatihan sangat besar.

2.4 Confusion Matrix

Confusion Matrix (matriks kebingungan) adalah suatu metode evaluasi hasil pemodelan klasifikasi *machine learning* yang dimana *output*nya dapat berupa dua kelas atau lebih. *Confusion Matrix* ini menyajikan tabel atau gambar hasil kombinasi berbeda dari nilai prediksi dan nilai aktual [27]. Adapun kegunaan dari *Confusion Matrix* ialah mengukur performa dari model klasifikasi *machine learning* yang diekstrak dari *dataset* pengujian yang telah diperoleh nilai kebenarannya (nilai aktual).

Terdapat empat nilai yang ditampilkan pada Tabel *Confusion Matrix,* yaitu *True Positive* (TP)*, False Positive* (FP)*, True Negative* (TN)*, False Negative* (FN)*.* Untuk lebih detail ditampilkan pada Tabel 2.1 [27].

Tabel 2. 1 *Confusion matrix.*

		Nilai Aktual	
		Positive	*Negative*
Nilai Aktual	*Positive*	TP	FP
	Negative	FN	TN

Keterangan [27]:

a. *True Positive* (TP): Jumlah data yang bernilai Positif dan diprediksi benar sebagai Positif.
b. *False Positive* (FP): Jumlah data yang bernilai Negatif tetapi diprediksi sebagai Positif.
c. *False Negative* (FN): Jumlah data yang bernilai Positif tetapi diprediksi sebagai Negatif.
d. *True Negative* (TN): Jumlah data yang bernilai Negatif dan diprediksi benar sebagai Negatif.

Pada penelitian ini, TP ditujukan untuk limbah kain bernilai *true* (benar) hasil klasifikasi, FP untuk kain yang digolongkan tidak sesuai klasifikasi, FN untuk jumlah kain salah hasil klasifikasi, dan TN untuk jumlah kain bukan bernilai benar dalam nilai aktual dan tidak bernilai benar pada hasil klasifikasi. Sebagai contoh terdapat sejumlah kain dengan tingkatan warna yang berbeda yaitu terang, dan gelap masing-masing satu buah kain, dalam proses klasifikasi tingkat kecerahan warna kain, kain terang diklasifikasi benar atau sesuai (artinya TP=1 buah, dan kain yang gelap hasil klasifikasi bernilai cerah (tidak sesuai). Maka, FP=1. Sedangkan pada penelitian ini TN dan FN bernilai 0, karena sitem klasifikasi tidak untuk mencari hasil negatif (tidak sesuai).

Confusion Matrix merupakan salah satu *tools* analitik prediktif yang menampilkan dan membandingkan nilai aktual atau nilai sebenarnya dengan nilai hasil prediksi model yang digunakan untuk menghasilkan matriks evaluasi, diantaranya:

a. *Accuracy* (akurasi) : menggambarkan seberapa akurat model dalam mengklasifikasikan dengan benar. Secara matematis ditulis sebagai berikut:

$$Akurasi = \frac{TP + TN}{TP + FP + FN + TN} \quad (2.5)$$

b. *Precision* : menggambarkan akurasi antara data yang diminta dengan hasil prediksi yang diberikan oleh model. Secara matematis ditulis sebagai berikut:

$$Precision = \frac{TP}{TP + FP} \quad (2.6)$$

c. *Recall* atau *sensitivy :* menggambarkan keberhasilan model dalam menemukan kembali sebuah informasi, Secara matematis ditulis sebagai berikut:

$$Recall = \frac{TP}{TP + FN} \quad (2.7)$$

d. *F1-score* : menggambarkan perbandingan rata - rata *Precision* dan *Recall* yang dibobotkan. Biasanya akurasi dijadikan acuan peformansi algoritma jika *dataset* memiliki data *False Negative* dan *False*

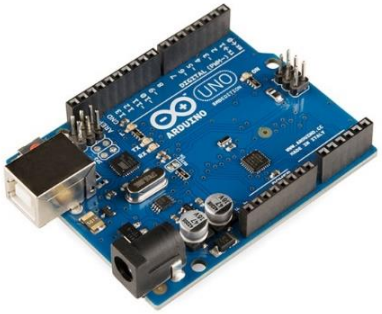
Positive yang sangat mendekati (*symmetric*). Namun jika jumlahnya tidak mendekati, maka sebaiknya menggunakan *F1-Score* sebagai acuan. Secara matematis ditulis sebagai berikut:

$$F1 - score = \frac{2\ x\ Precision\ x\ Recall}{Precision\ +\ Recall} \quad (2.8)$$

2.5 Arduino UNO

Arduino adalah pengendali mikro *singleboard* yang bersifat *open-source* yang dirancang untuk memudahkan penggunaan elektronik pada berbagai bidang. *Controller* Arduino menggunakan keluarga mikrokontroler Atmega yang dirilis oleh Atmel. Penggunaan *controller* Arduino menggunakan bahasa pemrograman tersendiri, namun masih memiliki kemiripan *syntax* dengan bahasa pemrograman C [29].

Arduino UNO merupakan papan mikrokontroler berbasis ATmega328 yang memiliki 14 pin *input* dari *output* digital dimana 6 pin *input* tersebut dapat digunakan sebagai *output* PWM dan 6 pin *input* analog, 16 MHz osilator kristal, koneksi USB, *jack power*, ICSP *header*, dan tombol reset. Arduino ini berisikan segala yang diperlukan untuk mendukung mikrokontroler, cukup dengan dihubungkan ke komputer menggunakan kabel USB atau diatur dengan adaptor AC ke DC atau baterai untuk memulai. Arduino UNO kompatibel dengan berbagai *shield* yang dirancang untuk Arduino *Duemilanove* atau *Diecimila.* Arduino UNO ditujukan pada Gambar 2.3 [29].



Gambar 2. 3 Arduino UNO.

2.6 Arduino IDE

Arduino merupakan sebuah platform yang terdiri dari *software* berupa Arduino IDE dan *hardware* berupa Arduino *Board*. Tujuan dari Arduino adalah pengguna akan dimudahkan dalam mempelajari pemrograman dan pembuatan suatu proyek-proyek elektronika maupun otomasi. Arduino menggunakan *software processing* yang digunakan untuk menuliskan program kedalam Arduino. *Processing* merupakan penggabungan antara bahasa C++ dan Java. Arduino IDE bersifat *open source* yang memungkinkan penggunanya dapat menggunakannya di berbagai *operating system* (OS) seperti *LINUX*, *Mac OS*, dan *Windows*. Pada *software* Arduino IDE mengkombinasikan *hardware*, bahasa pemrograman, dan *Integrated Development Environment* (IDE) yang canggih. IDE merupakan suatu *software* yang berperan dalam penulisan program, meng-*compile* menjadi kode biner, dan meng-upload ke dalam memori mikrokontroler [30].

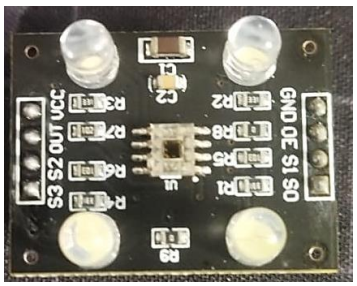
2.7 Sensor Warna TCS3200

Sensor warna TCS3200 adalah sensor yang digunakan pada aplikasi mikrokontroler untuk mendeteksi frekuensi warna merah (*red*), hijau (*green*) dan biru (*blue*) dari suatu obyek. Sensor ini menghasilkan keluaran berupa gelombang persegi (*duty cycle* 50%) dengan frekuensi yang berbanding lurus dengan intensitas cahaya [17].

Pada TCS3200, konverter intensitas cahaya ke frekuensi membaca 8x8 deret rangkaian fotodioda. 16 fotodioda memiliki filter merah, 16 fotodioda memili filter warna biru, 16 fotodioda memiliki filter hijau dan 16 fotodioda lainnya tidak memiliki filter [17][11].

Sejumlah penelitian menunjukan efektifitas penggunaan sensor ini untuk melakukan ekstraksi fitur warna (RGB) buah-buahan. Di temukan bahwa Sensor warna TCS3200 dapat digunakan untuk mendeteksi warna buah-buahan dengan tingkat kematangan yang berbeda-beda dengan jarak optimal 2 cm terhadap obyek. Adapun objek-objek yang digunakan pada penelitian tersebut antara lain buah tomat, pisang dan belimbing. Adapun beberapa faktor yang mempengaruhi akurasi sensor antara lain pencahayaan, jenis benda berwarna yang akan dideteksi serta

jarak antara sensor dengan obyek warna [11]. Sensor warna TCS3200 ditujukan pada Gambar 2.4 [13].



Gambar 2. 4 Sensor warna TCS3200.

2.8 *Load Cell* HX-711

Load Cell merupakan transduser yang mengubah gaya atau tekanan menjadi keluaran listrik . Besarnya *output* listrik ini berbanding lurus dengan gaya yang diterapkan. *Load cell* memiliki *strain gauge*, yang berubah bentuk ketika tekanan diberikan padanya. Dan kemudian *strain gauge* menghasilkan sinyal listrik pada deformasi karena perubahan resistansi efektif pada deformasi. *Load cell* biasanya terdiri dari empat pengukur regangan dalam konfigurasi jembatan *wheatstone*. Kemudian sinyal listrik yang dihasilkan oleh *Load cell* dalam beberapa milivolt perlu diperkuat oleh penguat HX711 *weighing sensor* [31]. Gambar sensor *load cell* dan penguat HX711 dapat dilihat pada Gambar 2.5 berikut.

Gambar 2. 5 *Loadcell* HX-711.

2.9 Buzzer

Buzzer adalah perangkat mekanis yang memproduksi suara melalu lengan mangnetik secara kontinyu pada diafragma. Perangkat ini dioperasikan dengan tegangan DC dan arus yang relatif kecil, secara umum membutuhkan 10mA. *Buzzer* menghasilkan suara 'berdengung' (nada tunggal) di frekuensi berkisar 300 hingga

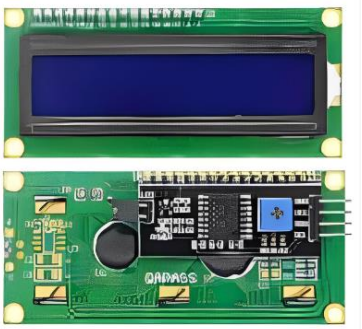
500 Hz. *Buzzer* adalah perangkat kecil dan dapat berupa salah satu panel terpasang atau PCB terpasang [32].

Buzzer dibedakan menjadi dua yaitu *buzzer* aktif dan *buzzer* pasif. *Buzzer* aktif bekerja dengan mengeluarkan bunyi peringatan tanpa perlu osilator eksternal atau rangkaian waktu (*timing)*. Sebaliknya *buzzer* pasif memerlukan osilator eksternal atau rangkaian waktu *(timing)*. *Buzzer* dapat dilihat pada Gambar 2.6 [32].

Gambar 2. 6 *Buzzer*.

2.10 LCD 16x2 *with* I2c

LCD *(Liquid Crystal Display*) adalah perangkat elektronik yang umum digunakan untuk menampilkan angka, teks, atau informasi lainnya dalam berbagai aplikasi. Pada sistem berbasis mikrokontroler, LCD 16x2 berperan sebagai antarmuka *output* utama. Layar ini menampilkan hasil dari nilai RGB yang dideteksi oleh sensor TCS 3200 dan berat yang dideteksi oleh sensor *Loadcell* HX711 dengan jelas, memudahkan pengguna dalam memantau data yang dihasilkan oleh sistem secara *real-time*. Penggunaan LCD 16x2 pada sistem ini sangat penting untuk memberikan informasi langsung dan akurat kepada pengguna. *Module LCD* 16x2 *with* I2c dapat dilihat pada Gambar 2.7 berikut [33].



Gambar 2. 7 *Module* LCD 16x2 with I2c.

2.11 *Servo* motor

Servo motor adalah salah satu jenis aktuator yang cukup banyak digunakan dibidang industri dan sistem robotika yang berfungsi untuk mendorong atau memutar objek dengan kontrol dengan presisi tinggi dalam hal posisi sudut, akselerasi, dan kecepatan, merupakan keunggulan yang tidak dimiliki oleh motor biasa [18]. Ketika presisi atau ketelitian pada mesin menjadi bagian terpenting dalam bidang industri untuk pemilihan *servo* motor. Performa tingkat akurasi (*high precision positioning*) dari motor *servo* ialah indikator utama spesifikasi. *Servo* motor dapat berputar dua arah, dimana arah dan sudut penggerak rotornya dapat dikendalikan dengan memberikan variasi lebar pulsa (*duty cycle*) sinyal PWM pada pin kontrolnya [11]. *Servo* motor disajikan pada Gambar 2.8 [11].

Gambar 2. 8 *Servo* motor.

2.12 Motor DC *Gearbox*

Motor DC dengan *gearbox* menggabungkan motor arus searah dan sistem roda gigi untuk mengatur torsi dan kecepatan. Motor DC mengubah energi listrik menjadi energi mekanik dengan efisiensi tinggi, sementara *gearbox* mengurangi kecepatan dan meningkatkan torsi. Sistem ini memungkinkan kontrol presisi yang dibutuhkan dalam aplikasi seperti robotika, kendaraan listrik, dan mesin industri [11]. Motor DC *gearbox* dapat dilihat pada Gambar 2.9 [11].

Gambar 2. 9 *Motor DC Gearbox*.

2.13 Conveyor Belt

Conveyor belt adalah sebuah alat atau mesin sederhana yang dirancang secara khusus untuk memudahkan perpindahan atau transportasi suatu barang atau material tertentu. Sistem *conveyor* ini diciptakan sebagai solusi agar terciptanya perpindahan atau transportasi barang secara efisien dibidang industri *Conveyor belt* dapat dioperasikan dengan kecepatan tertentu sesuai kebutuhan. Sistem kendali *conveyor* dapat menggunakan *switch on/off* yang sederhana, tipe *soft-soft* yang canggih dapat meredam beban saat dioperasikan, atau *driver* variabel frekuensi yang dapat mengontrol kecepatan motor DC [11]. *Conveyor belt* dapat dilihat pada Gambar 2.10 [11].

Gambar 2. 10 *Conveyor belt.*