# Pemanfaatan *Decision Tree* dan *Global Feature Extraction* pada Klasifikasi Motif Batik Indonesia

Mohammad Rifqi Farhansyah - 13521166[1]
*Program Studi Teknik Informatika*
*Sekolah Teknik Elektro dan Informatika*
*Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia*
*[1]13521166@std.stei.itb.ac.id*

***Abstract*—Pada era *big data, machine learning* menjadi pendekatan *data processing* yang cukup populer. Data yang diproses dapat berbentuk apapun, seperti teks, gambar, suara, video, maupun sinyal. *Machine learning* merupakan salah satu entitas yang dapat digunakan untuk menganalisis pola dari berbagai macam data tersebut. Pada makalah ini, akan direalisasikan metode *supervised machine learning (SML)* sebagai sebuah *data classifiers* dengan memanfaatkan salah satu konsep matematika diskrit, yaitu *decision tree* (J48). *Decision tree classifiers* merupakan salah satu metode klasifikasi data berbasis struktur *tree*. Data yang dipilih sebagai objek penulisan makalah ini adalah gambar. Namun demikian, dalam melakukan *image classifiers, decision tree* memerlukan beberapa komponen atribut sebagai paramater unik dari tiap *training object*. Parameter *image classification* makalah ini akan berbasis *Global Feature Extraction* yang melibatkan *Colour Histogram*, *Hu Moments*, dan *Haralick Texture*, sebagai nilai atribut unik untuk tiap *training image*. Makalah ini akan berfokus dalam melakukan *image classification* terhadap motif-motif batik Indonesia dengan memanfaatkan berbagai macam aspek tersebut.**

***Keywords*—Machine Learning, Classifiers, Global Feature Extraction, Decision tree.**

## I. PENDAHULUAN

Salah satu upaya yang dapat dilakukan untuk mendukung perkembangan Budaya Indonesia khususnya "batik" adalah dengan melakukan penelitian terhadap karakteristik tiap motif-nya. Batik dapat menjadi salah satu warisan leluhur yang bermarwah cukup tinggi disebabkan oleh seni klasik saat proses pembuatannya. Dewasa ini, permasalahan terkait pengakuan batik sebagai budaya oleh negara lain muncul karena kurangnya kesadaran warga negara Indonesia atas pelestarian batik itu sendiri. Motif-motif batik kurang begitu dikenali oleh masyrakat Indonesia pada umumnya, sehingga negara lain dengan mudah dapat melakukan sedikit modifikasi pada motif batik tertentu dan mengakuisisinya. Hal tersebut tentu perlu upaya pelestarian tertentu. Oleh karena itu, upaya pelestarian dengan mendokumentasikan seluruh hal terkait batik (motif, tekstur, dll) dapat menjadi solusi permasalahan yang tepat.

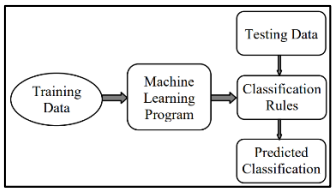
Dalam beberapa penelitian terbaru, batik hanya didokumentasikan sebagai sebuah kesatuan warna dan karakteristik bentuk-nya saja. Jurnal penelitian kurang begitu fokus pada karakteristik tekstur-nya, padahal hampir seluruh motif batik Indonesia memiliki simbol-simbol tertentu yang merepresentasikan budaya masing-masing. Oleh karena itu, pada makalah ini, penulis melibatkan komponen *supervised machine learning (SML)* dan *global feature extraction*. Agar pendokumentasian tersebut bisa lebih men-detail dan akurasinya tinggi.

*Supervised machine learning (ML)* merupakan salah satu metode *machine learning* yang memiliki tujuh algoritma berbeda sebagai sarana penyusunannya, antara lain : *decision table, random forest (RF) , naïve bayes (NB) , support vector machine (SVM), neural networks (Perceptron), JRip* dan *decision tree (J48)* yang menggunakan *waikato environment for knowledge analysis (WEKA).*

*Machine learning (ML)* sebenarnya merupakan sebuah cabang ilmu pengetahuan yang dapat dikategorikan sebagai bagian dari *artificial intelligence (AI)* karena algoritma-nya membentuk sebuah blok yang membuat kecerdasan komputer tidak hanya digunakan sebatas menyimpan dan mengambil data, tetapi menganalisis pola perilaku data-data tersebut. *Machine learning* telah mendapatkan inspirasi dari berbagai macam disiplin ilmu akademik lainnya, seperti ilmu komputer, statistika, biologi, dan psikologi, sehingga seringkali terdapat irisan antar displin ilmu tersebut. Inti dari *Machine Learning* adalah menemukan sebuah cara automasi untuk melakukan upaya prediksi berdasarkan pengalaman masa lalu. Pekerjaan tersebut telah berhasil dilakukan dengan baik oleh *classifiers*. *Classifiers* akan melakukan sebuah klasifikasi data dengan attribut yang bersesuaian kemudian melakukan analisis terhadapnya. Klasifikasi sendiri merupakan proses pembagian data ke dalam klas-klas tertentu dengan menggunakan dataset model untuk memprediksi *unknown values* (variabel keluaran) berdasarkan beberapa *known values* (variabel masukan), sebagai contoh: Pada proses klasifikasi yang dilakukan terhadap dataset D berisi beberapa objek adalah sebagai berikut:

- *Set size* $\rightarrow A = \{A_1, A_2, \dots, A|A|\}$, dengan $|A|$ sebagai jumlah dari atribut atau ukuran set A.
- *Class label* $\rightarrow C$: *Target attribute;* $C = \{c_1, c_2, \dots, c|C|\}$, *dengan* $|C|$ sebagai jumlah *class* dan $|C| \geq 2$.

Tujuan objektif dari *machine learning* adalah memproduksi fungsi prediksi/klasifikasi terkait nilai dari atribut di A dan *class* di C terhadap dataset D. Dalam algoritma *machine learning*, setiap entitas dari dataset direpresentasikan dengan suatu set fitur yang sama. Fitur dapat berupa data berkelanjutan, kategori, atau *binary*. Jika sebuah entitas dinyatakan dengan *known labels*, maka skema pembelajaran disebut sebagai *supervised machine learning (SML)*, sementara *unsupervised machine learning* melakukan pendekatakan tanpa label.




*Gambar 1.1 Arstitektur Klasifikasi*
*Sumber:https://pdfs.semanticscholar.org/7a89/00156b939d4377ff2e42af13618fed8ed443.pdf*

Dalam makalah ini, penulis akan lebih fokus terhadap pembahasan *machine learning* dengan model *supervised learning*. Sesuai dengan pemaparan di atas, jenis ini memerlukan data ber-label atau nilai tertentu yang akan diprediksi oleh sebuah model. Kemudian, data-data tersebut akan diseleksi agar mendapatkan data tanpa *defek* serta mencakup kebutuhan model untuk melakukan prediksi yang akurat.

*Tabel 1.1 Entitas dengan Known Labels*
*Sumber:https://pdfs.semanticscholar.org/7a89/00156b939d4377ff2e42af13618fed8ed443.pdf*

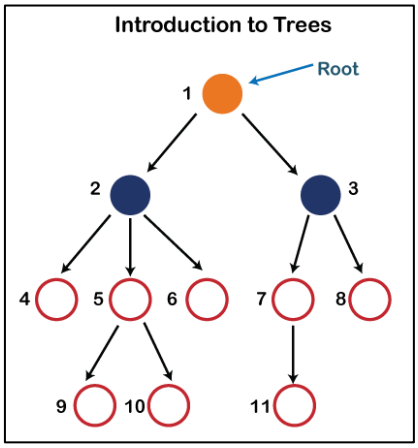
| Data in standard Format | | | | | |
|---|---|---|---|---|---|
| Case | Feature 1 | Feature 2 | … | Feature n | Class |
| 1 | aaa | bbb | … | nnn | Yes |
| 2 | aaa | bbb | … | nnn | Yes |
| 3 | aaa | bbb | … | nnn | No |
| … | … | … | … | … | … |

Selain itu, pada makalah ini, penulis akan menggunakan model *supervised machine learning* berbasis *decision tree* karena kompleksitasnya yang relatif rendah, tingkat akurasi yang cukup, serta kesesuaian penerapannya dengan materi pada mata kuliah IF2120 Matematika Diskrit.

## II. Landasan Teori

### A. Pengertian dari Struktur Data Tree

*Tree* / Pohon merupakan graf terhubung yang tidak mengandung sirkuit serta memiliki hierarki yang jelas di antara elemen-elemennya (berupa level atau tingkatan). Setiap elemen dalam *tree* (disebut juga sebagai *node*) mengandung nilai dan beberapa pointer ke elemen "anaknya".

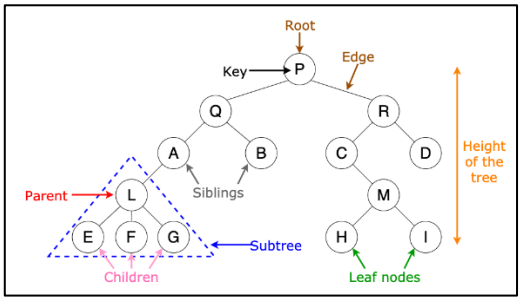



*Gambar 2.1 Contoh Struktur Data Tree*
*Sumber: https://static.javatpoint.com/ds/images/tree2.png*

Dalam beberapa poin, struktur data *tree* memiliki struktur yang mirip dengan struktur data *graph*. Hal ini karena *tree* merupakan sebuah *graph* dengan syarat tertentu. Beberapa properti dari *tree:*

1. Struktur data yang rekursif : *Tree* juga dikenal sebagai *recursive data structure*. Sebuah pohon dapat didefinisikan rekursif karena *root node* memiliki hubungan dengan semua *roots* dari tiap *subtrees*-nya.
2. Jumlah sisi : Apabila terdapat n simpul, maka akan terdapat n-1 sisi. Tiap panah di struktur menggambarkan hubungan masing-masing komponen.
3. Tanpa sirkuit : Apabila ditambahkan satu buah *edge*/sisi baru tanpa penambahan *node* baru, maka struktur data tersebut tidak akan menjadi *tree* karena terbentuknya sirkuit.
4. Bilangan kromatis : Sembarang tree memiliki bilangan kromatis dua (warna *node* antara orangtua dengan anak harus berbeda).

### B. Istilah dalam Struktur Data Tree




*Gambar 2.2 Terminologi pada Tree*
*Sumber:https://miro.medium.com/max/1950/1*PWJiwTxRdQy8A_Y0hAv5Eg.png*

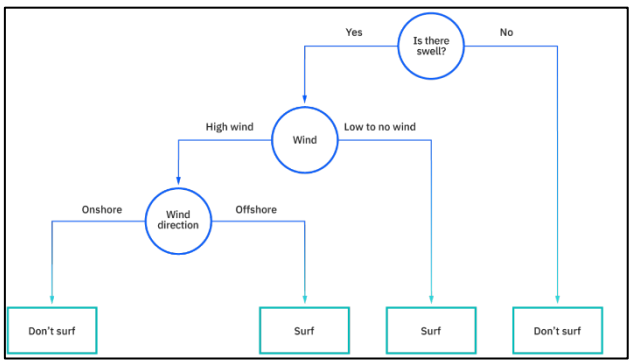
Beberapa istilah/terminologi pada struktur data *tree*, antara lain:

1. Anak (*child* atau *children*) dan Orangtua (*parent*) :
   E, F, dan G adalah anak-anak dari node L, L adalah orangtua dari anak-anak itu.
2. Lintasan (*path*) :
   Lintasan dari P ke E adalah P, Q, A, L, E. Panjang lintasan dari P ke E adalah 5.
3. Saudara kandung (*sibling*) :
   A adalah saudara kandung B, tetapi A bukan saudara kandung C, dikarenakan orangtua mereka berbeda.
4. Tingkatan (*level*) :
   Panjang lintasan dari *root node* ke *node* tersebut. Sebagai contoh, tingkatan dari node A adalah 2.
5. Tinggi (*height*) :
   Tinggi dari suatu *tree* adalah lintasan terpanjang dari *root node* menuju *leaf node*. Tinggi dari *tree* pada contoh adalah 4.
6. Upapohon (*subtree*) :
   Upapohon (L (E () F () G ())) merupakan upapohon dari pohon pada contoh.
7. Derajat (*degree*) :
   Derajat sebuah *node* adalah jumlah dari upapohon atau anak pada node tersebut. Sebagai contoh, derajat dari L adalah 3 dan derajat dari R adalah 2.

8. Daun (*leaf*) :
   Daun adalah *node* yang berderajat nol (tidak mempunyai anak). Sebagai contoh, E, F, G, B, H, I, dan D adalah daun dari pohon.
9. Simpul dalam (*internal nodes*) :
   Simpul dalam adalah *node* yang mempunyai anak (berderajat lebih dari nol). Sebagai contoh, P, Q, dan A adalah simpul dalam.

## C. Definisi dari Decision Tree

*Decision tree* merupakan non-parametrik *supervised learning algorithm* yang memanfaatkan pendekatan klasifikasi dan regresi secara bersamaan. *Decision tree* bekerja dengan mencari cara untuk membagi data menjadi beberapa bagian tertentu, kemudian melakukan optimalisasi terhadap masing-masing parameter. *Decision tree* biasa digunakan untuk melakukan pengklasifikasian suatu data diskrit, tetapi *decision tree* juga dapat digunakan untuk regresi data kontinu. *Decision tree* juga memiliki kecenderungan untuk *overfitting*, dimana akurasi model bernilai tinggi hanya ketika menggunakan data *training*, tetapi ketika dimasukkan data baru ke dalam *database*, hasil akan menjadi tidak akurat. Hal ini disebabkan oleh *decision tree* yang mencoba membuat aturan baru hasil inferensi terhadap data *training*. Oleh karena itu, data *training* yang dimasukkan sebagai sebuah model haruslah universal dan seimbang (frekuensi data tersebar dengan rata). Berikut ini adalah contoh pembentukan *decision tree* dengan menggunakan analisis inferensi data masukan, misalkan terdapat beberapa masukan yang berisi data arah angin, kelembapan udara, dll, sedangkan pertanyaannya adalah "Apakah hari ini bisa bermain papan seluncur?".




*Gambar 2.3 Contoh Penerapan Decision Tree*
*Sumber: https://www.ibm.com/content/dam/connectedassets-adobe-cms/worldwide-content/cdp/cf/ul/g/10/3c/Decision-Tree-Example.component.xl.ts=1640801899950.png/*

Selain menginferensikan data, *decision tree* juga dapat digunakan sebagai representasi fungsi boolean dalam bentuk apapun. Hal ini dilakukan dengan merepresentasikan *atomic* pada *node* dan nilai boolean dari *atomic* tersebut (*true* atau *false*) pada *edge decision tree*. Dengan begitu, *decision tree* dapat memodelkan data secara lebih komprehensif dibandingkan analisis regresi.

## D. Algoritma Pembentukan Decision Tree

Dalam pembentukan sebuah prediksi/klasifikasi dari sebuah *dataset* menggunakan decision tree, algoritma yang diterapkan bekerja secara *top-down*, dimulai dari *root node* struktur tersebut. Algoritma ini akan membandingkan nilai dari *root attribute* dengan *real dataset attribute*, kemudian perbandingan nilai akan terus dilakukan melalui tiap *branch* dari struktur data *tree*. Pada *node* selanjutnya algoritma akan membandingkan *attribute value* dengan *sub-nodes* yang lain. Secara detail, algoritma yang dilakukan sebagai berikut:

1. Algoritma dimulai dari *root node* struktur data *tree* tertentu (S).
2. Temukan *attribute* terbaik dari sebuah *dataset* menggunakan *Attribute Selection Measure* (*ASM*).
3. Bagi S menjadi beberapa bagian *subsets* yang mengandung nilai dari *best attributes*.
4. Bentuk sebuah *decision tree node* yang mengandung *best attribute*.
5. Secara rekursif, bentuklah kembali sebuah *decision tree* baru dengan menggunakan *subsets* dari *dataset* yang dibuat pada tahap ke-3. Lakukan langkah ini secara terus menerus hingga proses mencapai sebuah *final node* atau *leaf node*.

Selain itu, pada makalah ini juga akan diberikan beberapa komponen yang diperhitungkan agar bisa didapatkan struktur data *tree* terbaik menggunakan *Attribute Selection Measure*.

1. *Information Gain*
   *Entropy* merupakan salah satu perhitungan terkait tingkat kemurnian suatu input set. Dalam fisika dan matematika, *entropy* berkaitan dengan tingkat persebaran data atau kemurnian sistem. Pada ilmu komputer, *entropy* merujuk pada tingkat kemurnian pada beberapa grup berisi contoh model. *Information gain* sendiri merupakan nilai perubahan *entropy* awal dan akhir. *Information gain* akan menghitung perbedaan antara *entropy* sebelum dan *entropy* rata-rata setelah *splitting process* berdasarkan *attribute value*-nya.

$$Info(D) = -\sum_{i=1}^{m} pi \log_2 pi \qquad 2.1.1$$
$$Info_A(D) = \sum_{j=1}^{V} \frac{|Dj|}{|D|} \times Info(D_j) \qquad 2.1.2$$
$$Gain(A) = Info(D) - Info_A(D) \qquad 2.1.3$$

*Persamaan 2.1 Iterative Dichotomiser pada Decision Tree*

   dengan $Info(D)$ adalah rata-rata dari informasi yang dibutuhkan untuk melakukan identifikasi pada *class label* dari *tuple* pada D, $|D_j|/|D|$ merupakan nilai dari partisi ke-j, serta $Info_A(D)$ merupakan ekspektasi informasi yang dibutuhkan untuk mengklasifikasikan *tuple* D berdasarkan partisiasi dari A.
2. *Gain Ratio*
   *Information Gain* terbiaskan oleh *attribute* yang memiliki banyak solusi, sehingga diperlukan *attribute* dengan jumlah data yang cukup besar dan nilai tertentu. Oleh karena itu, diperlukan sebuah *gain ratio* yang akan mengatasi bias dengan menormalisasi *information gain* menggunakan *split info*.

$$SplitInfo_A(D) = -\sum_{j=1}^{V} \frac{|Dj|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right) \qquad 2.2.1$$
$$GainRatio(A) = \frac{Gain(A)}{SplitInfo_A(D)} \qquad 2.2.2$$

*Persamaan 2.2 J48 Algorithm for Gain Ratio*

   dengan $|D_j|/|D|$ merupakan nilai dari partisi ke-j dan v merupakan jumlah *discrete values* pada attribute A.

3. *Gini Index*

*Attribute selection measure* juga dapat diselesaikan menggunakan metode *Gini* untuk membuat *split point*-nya. *Gini Index* mempertimbangkan *binary split* untuk tiap attribut, sehingga dapat dihitung nilai kemurnian dari tiap partisi. Pada kasus *discrete-valued attributes*, subset yang memberikan nilai *gini index* minimum akan dipilih sebagai *splitting attribute*. Sementara pada kasus *continuous-valued attributes*, strategi yang digunakan adalah dengan memilih pasangan value yang bertetangga sebagai *possible split-point*, kemudian *point* dengan *gini index* yang lebih kecil akan dipilih sebagai *splitting point*.

$$Gini(D) = 1 - \sum_{i=1}^{m} Pi^2 \quad 2.3.1$$

$$Gini_A(D) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2) \quad 2.3.2$$

$$\Delta Gini(A) = Gini(D) - Gini_A(D) \quad 2.3.3$$

*Persamaan 2.3 Gini Index pada Decision Tree*

Namun demikian, *decision tree* juga dapat dibentuk menggunakan dua konsep algoritma lain, yaitu *greedy algorithm* dan *divide and conquer*. *Greedy Algorithm* merupkan algoritma yang membentuk solusi langkah per langkah dengan mencari nilai maksimum sementara pada setiap langkahnya. Algoritma ini disebut sebagai algoritma yang 'rakus' karena melakukan pemecahan masalah secara langsung tanpa mempertimbangkan sudut pandang lainnya. Sementara itu, *divide and conquer* merupakan suatu cara atau metode penyelesaian masalah dengan cara membagi masalah menjadi beberapa submasalah yang lebih sederhana, kemudian menyelesaikan masalah tersebut, lalu menggabungkannya kembali menjadi sesuatu yang umum.

### *D. Algoritma Pembentukan Global Feature Extraction*

*Feature extraction* merupakan tahapan awal yang berfokus pada pengenalan objek tertentu dan menempatkannya pada *class* yang sesuai. Proses ini dapat dibagi menjadi 2 proses utama: *Feature Selection* dan *Classification*. *Feature Selection* sangat penting dilakukan karena sebuah *classifiers* tidak akan mampu melakukan pengenalan terhadap data secara langsung. Kriteria dalam memilih fitur adalah ketercakupan seluruh informasi yang membedakan *class* satu dengan yang lain, sangat sensitif terhadap variabel yang tidak bersesuaian dengan masukan, serta terbatas jumlahnya, agar proses yang dilakukan dapat se-efisien mungkin.

*Feature extraction* memiliki peran yang sangat penting pada tahapan konstruksi dari klasifikasi sebuah motif tertentu karena hasilnya bertujuan untuk memberikan paramater klasifikasi pada tiap *class*. Pada proses ini, fitur yang sesuai akan diubah menjadi sebuah *feature vectors*. *Feature Vectors* ini kemudian akan digunkan oleh *classifiers* untuk mengenali input dengan target unit keluaran. Feature extraction pada dasarnya akan menentukan set parameter yang mendefinisikan bentuk dari karakter secara tepat dan unik. Tujuan mayor dari *feature extraction* adalah membentuk sebuah set fitur yang memaksimalkan *recognition rate* dengan jumlah elemen yang paling sedikit serta membentuk set fitur yang mirip untuk variasi entitas lainnya.

Dalam penentuan *features* yang sesuai dengan permasalahan ini, penulis menganalisa bahwa akan digunakan *features global image* agar mencakup kuantisasi keseluruhan *pixel image*, seperti warna, tekstur, bentuk, dan lain-lain. Beberapa contoh algoritma *global features extraction* adalah *color channel statistics, color histogram, hu moments, zernike moments, haralick textures, local binary pattern, histogram of oriented gradients,* dan *threshold adjacency statistics*. *Global feature extraction* yang akan diterapkan pada makalah ini adalah *colour histogram* untuk mengkuantisasi warna pada proses *image processing*; *haralick textures* untuk mengkuantisasi bentuk suatu objek tertentu; serta *hu moments* untuk mengkuantisasi tekstur dari masukan gambar. Selanjutnya, hasil dari *features extraction* akan diproses ke dalam sebuah vektor sebagai suatu parameter pada struktur *decision tree* yang dibentuk.

## III. IMPLEMENTASI

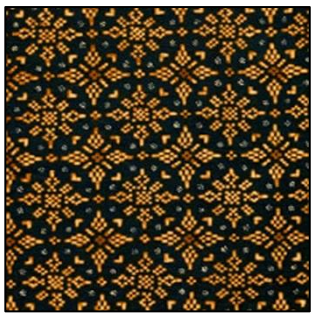
### *A. Dataset*

Pada makalah ini, penulis melakukan implementasi dengan menggunakan empat dataset motif batik khas Indonesia, yaitu: Ceplok, Lereng, Nitik,dan Parang. Tiap *labels* (dalam hal ini jenis-jenis motif) akan diterapkan 80 dataset serta satu gambar sebagai *test case* (gambar yang berbeda dari dataset).

Alasan pemilihan keempat dataset motif batik tersebut adalah petimbangan terkait keunikan tiap motif, tingkat kearifan yang terkandung dalam tiap motif, serta popularitas masing-masing motif batik. Dalam dataset juga diberikan beberapa motif yang merupakan hasil akulturasi dari motif satu dengan yang lain, misalnya: gabungan motif batik parang dan ceplok, nitik dan lereng, serta gabungan dari tiga motif sekaligus. Hal tersebut merupakan upaya untuk melakukan test pada *attributes* yang dibentuk berdasarkan dataset.

*Gambar 3.1 Gambar Batik Ceplok*
*Sumber:Dokumen Penulis*

*Gambar 3.2 Gambar Batik Lereng*
*Sumber:Dokumen Penulis*



*Gambar 3.3 Gambar Batik Nitik*
*Sumber:Dokumen Penulis*

*Gambar 3.4 Gambar Batik Parang*
*Sumber: Dokumen Penulis*

## *B. Struktur Program*

Directory Tree dari source code implementasi program adalah sebagai berikut:

```bash
    create_dataset.py
    formatting_output.py
    main.py
    README.md
    training_dataset.py

├──dataset
│   ├──test
│       1.jpg
│       2.jpg
│       3.jpg
│       4.jpg
│
│   └──train
│       ├──Ceplok
│       ├──Lereng
│       ├──Nitik
│       └──Parang
│
├──processing
    data.h5
    labels.h5

└──__pycache__
```

*Gambar 3.5 Directory Tree dari Source Code*
*Sumber:Dokumen Penulis*

Program terdiri atas empat sub-program yang berupa *python source code file*, folder *dataset* yang mencakup folder *'train'* sebagai lokasi penyimanan kumpulan gambar dataset serta folder '*test'* sebagai lokasi penyimpanan kumpulan gambar yang akan di-*test* pengklasifikasiannya, Selan itu, terdapat folder '*processing'* berisi dua *source file* dengan ekstensi file HDF5 yang merupakan hasil pemrosesan dengan algoritma-algoritma terkait. Untuk lebih lengkapnya, *source code* dapat diakses melalui pranala berikut ini.

## *C. Modul Eksternal*

Pada implementasi program digunakan beberapa modul eksternal, seperti *OpenCv2*, *numpy*, *sklearn*, *h5py*, *mahotas*, dll. *OpenCv2* merupakan modul eksternal yang digunakan dalam melakukan fungsi/prosedur terkait *image processing* (versi yang digunakan 4.5.4*). Modul OpenCv2* digunakan untuk membaca masukan gambar, melakukan *display* gambar, serta melakukan beberapa operasi fundamental pada gambar. *Numpy* merupakan modul eksternal yang dapat digunakan sebagai sarana pengolahan array agar lebih efisien dan efektif tanpa harus membentuk fungsi atau prosedur tambahan lainnya (versi yang digunakan 1.21.3). *Sklearn* merupakan modul eksternal yang sering digunakan dalam penerapan *supervised machine learning*, terutama dalam proses *classification* (versi yang digunakan 1.0.1). *Mahotas* juga merupakan salah satu modul yang terkait dengan *computer vision* dan *image processing* pada bahasa pemrograman *python*. Pada implementasi makalah ini, *mahotas* digunakan sebagai sarana untuk mengambil salah satu fungsi terkait penerapan *global features extraction*, yaitu: *haralick textures* (versi yang digunakan 1.4.13). Modul kelima yang diimpor adalah *h5py*, *library* eksternal yang dapat digunakan sebagai sarana pengolahan *HDF5 binary data format* (versi yang digunakan 3.6.0).

Selain itu, terdapat dua modul eksternal yang diimpor sebagai fasilitas pengolahan decision tree serta melakukan visualisasi. Modul pertama yang digunakan adalah *pydot* atau *pydotplus* (untuk *python* dengan versi di atas 3.9). *Pydot* digunakan sebagai penghubung agar bisa menjalankan modul kedua, yaitu: *graphviz*. *Graphviz* merupakan sebuah *open-source python module* yang dapat digunakan untuk membuat objek *graf/tree*. Modul ini menggunakan basis *DOT language* dari *Graphviz software.*

## *D. Implementasi Subprogram Create Dataset*

*Source file* create_dataset.py mengandung beberapa variabel *path*, *HDF5 source file*, serta nilai yang digunakan secara global. Selain itu, dalam create_dataset.py terdapat beberapa implementasi fungsi terkait *Global Features Extraction*, yaitu fitur *Hu Moments*, *Haralick Texture*, dan *Colour Histogram*. *Hu Moments* diimplementasikan dengan menggunakan modul eksternal *OpenCV2*. *Haralick Texture* diimplementasikan dengan menggunakan modul eksternal *mahotas* (seperti yang telah disebutkan di atas). *Colour Histogram* pada makalah ini diterapkan menggunakan pemanggilan fungsi dalam modul eksternal *OpenCV2*.

```python
def haralick_texture(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    haralick = mahotas.features.haralick(gray).mean(axis=0)
    return haralick
```

*Gambar 3.6 Implementasi fungsi Haralick Texture*
*Sumber: Dokumen Penulis*

```python
def hu_moments(image):
    image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    feature = cv2.HuMoments(cv2.moments(image)).flatten()
    return feature
```

*Gambar 3.7 Implementasi fungsi Hu Moments*
*Sumber: Dokumen Penulis*

```python
def colour_histogram(image):
    image = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    hist  = cv2.calcHist([image], [0, 1, 2], None, [bins, bins, bins], [0, 256, 0, 256, 0, 256])
    cv2.normalize(hist, hist)
    return hist.flatten()
```

*Gambar 3.8 Implementasi fungsi Colour Histogram*
*Sumber: Dokumen Penulis*

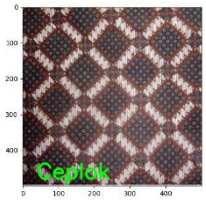
### *E. Implementasi Subprogram Training Datase*

*Source file* training_dataset.py mengandung variabel-variabel terkait *path, HDF5 source file,* serta nilai-nilai yang digunakan dalam pembentukan vektor dari *global features extraction.* Selain itu, terdapat dua buah implementasi fungsi/prosedur, yaitu: *train_model* dan *test_model.*

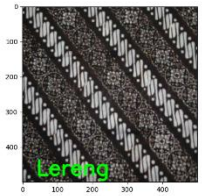
Fungsi *train_model* melakukan implementasi yang diawali dengan membaca *HDF5 source file* (data.h5 dan labels.h5). Tahapan selanjutnya adalah dengan melakukan *concat* atau penggabungan terhadap vektor-vektor hasil dari *features extraction.* Fungsi ini juga akan melakukan print data-data terkait dengan vektor-vektor pada proses *global features extraction*.

Fungsi yang kedua adalah *test_model* yang akan digunakan untuk implementasi klasifikasi suatu gambar terhadap atribut-atribut tertentu ke dalam beberapa *class* yang telah dibentuk. Fungsi ini kemudian akan melakukan *display image* terhadap tiap *test case image* satu per satu. Ketika melalukan *display* gambar akan ditampilkan *labels name* berdasarkan *class* yang sesuai dengan *test case image. Labels name* akan berada pada koordinat (40,480) dan berwarna hijau.

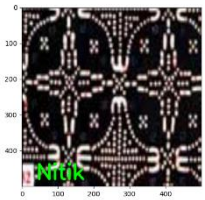
Berikut ini contoh keluaran gambar hasil *classification* dengan penerapan *decision tree* dan *global features extraction*:



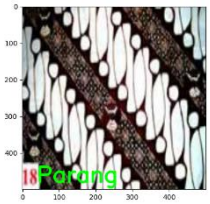

*Gambar 3.9 Klasifikasi Motif Batik Ceplok*
*Sumber: Dokumen Penulis*




*Gambar 3.10 Klasifikasi Motif Batik Lereng*
*Sumber: Dokumen Penulis*



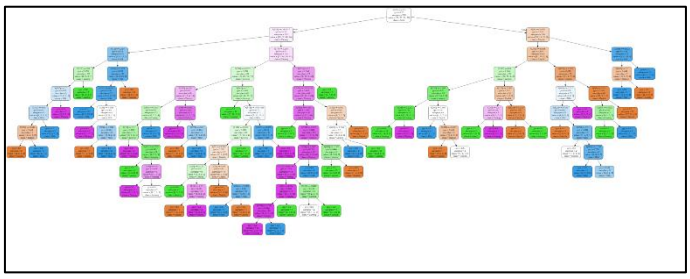

*Gambar 3.11 Klasifikasi Motif Batik Nitik*
*Sumber: Dokumen Penulis*




*Gambar 3.12 Klasifikasi Motif Batik Parang*
*Sumber: Dokumen Penulis*

### *D. Implementasi Subprogram Formatting Output*

*Source file* formatting_output.py mengandung deklarasi fungsi untuk melakukan *export* struktur data *decision tree* hasil klasifikasi dalam bentuk DOT File (ekstensi yang digunakan oleh *graphviz* untuk menampilkan *output*). Hasil *export* dari *decision tree* akan bernama *'tree.dot'* dan untuk memprosesnya menjadi sebuah file gambar diperlukan bantuan dari *WebGraphviz* yang dapat diakses melalui pranala berikut ini. Berikut ini adalah contoh hasil *decision tree* dari klasifikasi motif batik dengan bantuan *global feature extraction.*

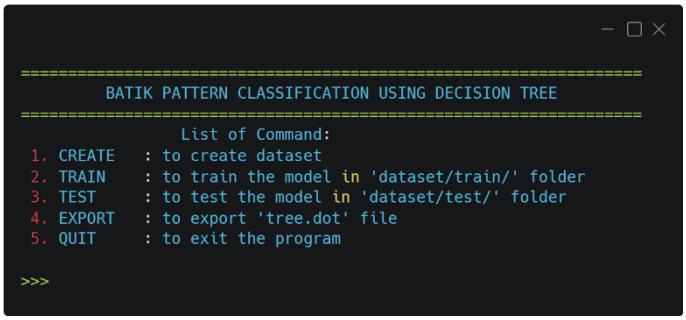


*Gambar 3.13 Visualisasi Decision Tree Hasil Klasifikasi Motif Batik*
*Sumber: Dokumen Penulis*

### *E. Implementasi Program Utama*

*Source file* main.py mengandung daftar instruksi yang dapat dimasukkan ke dalam program. Instruksi yang diterima oleh program adalah *CREATE, TRAIN, TEST, EXPORT, dan QUIT.* Instruksi *CREATE* digunakan untuk melakukan *trigger* terhadap subprogram *create_dataset,* yang bertujuan untuk melakukan *image and data processing* terhadap *image training dataset*. Selanjutnya, instruksi *TRAIN* akan melakukan eksekusi terhadap subprogram *training_dataset* yang bertujuan untuk membuat *decision tree models* hasil klasifikasi berdasarkan beberapa parameter atribut tertentu. Selanjutnya, instruksi *TEST* akan melakukan eksekusi terhadap fungsi *test_model* yang bertujuan untuk melakukan prediksi terhadap klasifikasi tertentu berdasarkan kemiripan atribut-nya. Instruksi *EXPORT* dapat digunakan untuk mencetak '*tree.dot'* yang merupakan *DOT File* dari *decision tree* hasil instruksi *CREATE, TRAIN,* dan *TEST*. Selanjutnya, instruksi terakhir yang dapat dimasukkan adalah *QUIT,* untuk melakukan interupsi pada program. Beberapa ketentuan yang terkait dengan program, antara lain:

1. Instruksi harus dilakukan secara urut, mulai dari *CREATE, TRAIN, TEST, EXPORT,* dan *QUIT.*
2. Gambar dalam tiap folder harus berjumlah 80 sesuai dengan deklarasi nilai ketika *prepare_data.*
3. Nama folder pada '*dataset/train*' akan digunakan sebagai *labels name* untuk tiap *class.*




*Gambar 3.14 Tampilan Program Utama*
*Sumber: Dokumen Penulis*

## IV. PENGOLAHAN DATA

Implementasi dalam makalah ini melibatkan beberapa perhitungan matematis, sehingga didapatkan beberapa data hasil ujicoba terhadap gambar *training*. Data-data perhitungan yang didapatkan dari klasifikasi motif batik menggunakan *decision tree* dengan *global features extraction*, antara lain:

### A. Data Subprogram Create Dataset

*Tabel 4.1 Data Fungsi Prepare Data*
*Sumber:Dokumen Penulis*

| No | Parameter | Nilai |
|---|---|---|
| 1 | *Feature Vector Size* | (320,532) |
| 2 | *Training Labels* | (320,) |
| 3 | Target Labels Shape | (320,) |

### B. Data Subprogram Training Dataset

*Tabel 4.2 Data Fungsi Train Models dan Test Models*
*Sumber:Dokumen Penulis*

| No | Parameter | Nilai |
|---|---|---|
| 1 | *Features Shape* | (320,532) |
| 2 | *Labels Shape* | (320,) |
| 3 | Train Data | (288,532) |
| 4 | Test Data | (32,532) |
| 5 | Train Labels | (288,) |
| 6 | Test Labels | (32,) |
| 7 | Accuracy | 56,25 % |

## V. KESIMPULAN

*Decision tree* merupakan salah satu model *supervised machine learning* yang digunakan untuk melakukan prediksi/ klasifikasi. *Decision tree* memiliki sebuah struktur data rekursif dalam bentuk *tree* yang diawali sebuah *root node* dan terdiri atas *edge* yang merepresentasikan semua kemungkinan relasi antar *node* dalam *tree* sebagai representasi atas parameter data tertentu. *Decision tree* dapat melakukan *prediction/ classification* terhadap beberapa jenis data, seperti *boolean*, *numerik*, *string*, maupun vektor. Salah satu algoritma pembentukan *decision tree* adalah dengan membandingkan nilai dari *root attribute* terhadap *real dataset attribute*, kemudian perbandingan nilai akan terus dilakukan melalui tiap *branch* dari struktur data *tree*. Selain itu, *decision tree* juga dapat dibentuk menggunakan *greedy algorithm* dan *divide and conquer algorithm. Decision tree* dapat dikaitkan dengan topik *image processing* melalui penerapan klasifikasi berdasarkan *global features extraction. Features extraction pada makalah ini* diterapkan menggunakan parameter *hu moments, haralick Texture, dan colour histogram*. Implementasi program secara keseluruhan pada komputer dapat dilakukan dengan menggunakan beberapa modul eksternal, seperti : *OpenCv2*, *numpy*, *sklearn*, *h5py*, *mahotas*, dll.

## VI. UCAPAN TERIMA KASIH

Puji dan syukur penulis panjatkan kepada Tuhan Yang Maha Esa atas segala rahmat dan kasih karunia-Nya yang telah memberikan kesehatan dan kesempatan kepada penulis sehingga penulis dapat menyelesaikan makalah ini. Penulis juga mengucapkan terima kasih sebesar-besarnya kepada seluruh pihak yang telah membantu penulis dalam menyelesaikan makalah ini, antara lain:

1. Ibu Fariska Zakhralativa, S.T, M.T. sebagai dosen pengampu dalam mata kuliah IF2120 Matematika Diskrit K02.
2. Dr. Ir. Rinaldi Munir, M.T. atas kontribusi buku dan materi yang penulis kutip pada makalah ini.
3. Seluruh pihak yang telah membuat source code dan modul secara *open-source* serta penulis kutip pada makalah ini.

## REFERENCES

[1] Jazuli Hafidz. 2018. *Using Decision Tree Method for Car Selection Problem.* https://medium.com/machine-learning-guy/using-decision-tree-method-for-car-selection-problem-5272675451f9, diakses pada 6 Desember 2022.

[2] Bahzad Taha Jijo, Adnan Mohsin Abdulazees. 2021. *Classification Based on Decision Tree Algorithm for Machine Learning*. https://jastt.org/index.php/jasttpath, diakses pada 6 Desember 2022.

[3] Rinaldi Munir. 2005. *Matematika Diskrit edisi 3*. https://www.academia.edu/29914530/Matematika_Diskrit_RInaldi_Munir, diakses pada 9 Desember 2022.

[4] Kenneth H.Rosen. 2013. *Discrete Mathematics and Its Application.* https://www.goodreads.com/book/show/1800803.Discrete_Mathematics_and_Its_Applications, diakses pada 6 Desember 2022.

[5] Abdul Haris Rangkuti, Zulfany Erlisa Rasjid, dan Djunaidi Santoso. 2015. *Batik image classification using treeval and treefit as decision tree function in optimizing content based batik image retrieval*. https://www.sciencedirect.com/science/article/pii/S1877050915020803, diakses pada 6 Desember 2022.

[6] Iqbal Muhammad, Zhu Yan. 2015. *Supervised Machine Learning Approaches:ASurvey.*https://pdfs.semanticscholar.org/7a89/00156b939d4377ff2e42af13618fed8ed443.pdf, diakses pada 9 Desember 2022.

[7] Han Liu, Mihaela Cocea, Well Ding. 2017. *Decision tree learning based feature evaluation and selection for image classification*. https://ieeexplore.ieee.org/document/8108975, diakses pada 6 Desember 2022.

[8] Osisanwo F.Y, Akinsola J.E.T, Awodele O, Hinmikaiye J.O. 2017. *Supervised Machine Learning Algorithms: Classification and Comparison*. https://www.researchgate.net/profile/J-E-T-Akinsola/publication/318338750_Supervised_Machine_Learning_Algorithms_Classification_and_Comparison/links/596481dd0f7e9b819497e265/Supervised-Machine-Learning-Algorithms-Classification-and-Comparison.pdf, diakses pada 9 Desember 2022.

[9] Hanung Risqiwidianto. 2021. *Deep Learning Batik Classificatiom.* https://github.com/hanungrisqiwidianto/klasifikasi_batik, diakses pada 9 Desember 2022.

[10] Rinaldi Munir. 2022. *IF2120 Matematika Diskrit – Semester I tahun 2022/2023.* https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2022-2023/matdis22-23.htm#Makalah, diakses pada 9 Desember 2022.

[11] JavaTPoint. 2022. *Tree Data Structure.* https://www.javatpoint.com/tree, diakses pada 9 Desember 2022.

[12] IBM. 2022. *Decision Trees.* https://www.ibm.com/topics/decision-trees, diakses pada 9 Desember 2022.

[13] GeeksForGeeks. 2019. *Python OpenCV | cv2.putText() method with Basic*. https://www.geeksforgeeks.org/python-opencv-cv2-puttext-method/, diakses pada 9 Desember 2022.

[14] Priya Pedamkar. 2020. Data Structures Tutorial | *Decision Tree Algorithm.* https://www.educba.com/decision-tree-algorithm/, diakses pada 9 Desember 2022.

[15] Viverk Kumar. 2022. *Decision Tree Algorithm overview explained.* https://towardsmachinelearning.org/decision-tree-algorithm/, diakses pada 9 Desember 2022.

[16] Avinash Navlani. 2018. *Decision Tree Classification in Python Tutorial*. https://www.datacamp.com/tutorial/decision-tree-classification-python, diakses pada 9 Desember 2022.

[17] Gaurav Kumar. 2014. *A Detailed Review of Feature Extraction in Image Processing Systems* https://www.researchgate.net/publication/260952140_A_Detailed_Review_of_Feature_Extraction_in_Image_Processing_Systems, diakses pada 9 Desember 2022.

[18] Jianjian Yan. 2020. *A hybrid scheme-based one-vs-all decision trees for multi-class classification tasks*. https://www.sciencedirect.com/science/article/abs/pii/S0950705120302598?casa_token=k3TFgg11BaoAAAAA:vVGyP-a0nQtplVtBDLvHkeyenUfKOGizEVeB7tVsLahVY6foZXoqGvMcBdfIB4BCWTGcmykZJn4, diakses pada 10 Desember 2022.

[19] GeeksForGeeks. 2022. *Decision Tree Introduction with example*. https://www.geeksforgeeks.org/decision-tree-introduction-example/?ref=rp, diakses pada 10 Desember 2022.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 9 Desember 2022

Mohammad Rifqi Farhansyah 13521166