**INTRO**

Aplikasi ini menggunakan teknologi *Machine Learning*. ML merupakan salah satu cabang dari disiplin ilmu Kecerdasan Buatan (Artificial Intellegence) yang membahas mengenai pembangunan sistem yang berdasarkan pada data. Teknik pengaplikasian ML yang digunakan di sini adalah supervised learning. *Supervised learning* adalah salah satu metode ML yang mempelajari data yang sudah diberi label sehingga dapat menemukan pola atau fungsi dari data tersebut yang nantinya akan dijadikan sebahai model prediksi.

OpenCV (Open Source Computer Vision Library) adalah sebuah *software library* yang bersifat *open source* untuk *Computer Vision* dan ML. OpenCV dikembangkan untuk menyediakan infrastruktur pokok dalam pengolahan aplikasi berbasis citra pada komputer. Aplikasi ini menggunakan *OpenCV* sebagai *library* dasar penerapan algoritma ML.

Bahasa pemrograman yang digunakan adalah Python.

**ALGORITMA**

**Proses *Training***

Berikut adalah proses yang bertujuan untuk melatih model supaya dapat mengenali wajah, terdapat folder images dan di dalamnya terdapat sub-folder yang di dalamnya terdapat foto-foto dari orang yang sama, nama sub-folder itu akan digunakan sebagai label. Proses ini dijalankan oleh face-train.py

1. Menentukan letak gambar dan mengambil nama sub-folder sebagai *string* yang nantinya akan dipakai sebagai nama label
2. Mendeteksi jumlah foto yang ada pada sub-folder tersebut kemudian memberikan *indexed\_id* untuk proses training
3. Membuka setiap gambar kemudian menentukan bagian wajah (*Region of Interest area / ROI area*)
4. Memproses dan mengubah bagian wajah tersebut ke dalam bentuk array untuk dipelajari polanya
5. Setelah semua gambar selesai diproses, kemudian dilakukan tahap pembelajaran untuk mendapatkan pola klasifikasi pengidentifikasi wajah
6. Setelah selesai proses *training,* kemudian simpan model pada *instance* ini untuk digunakan pada proses berikutnya

**Proses Implementasi model**

Proses ini dijalankan oleh faces.py

1. Komputer menerima *input* citra via *webcam*
2. Menampilkan citra tersebut pada *screen* Monitor dalam sebuah *live view* *window*
3. Mendeteksi wajah yang ada pada citra tersebut
4. Membuat persegi (*rectangle*) sebagai penanda jika ada wajah telah terdeteksi (*Region of Interest* *area* / *ROI area*)
5. Memotong (*crop*) bagian wajah tersebut kemudian menampilkan pada *window* lain
6. Menggunakan hasil pembelajaran (*trained model*) dari proses *training* untuk mengidentifikasi identitas wajah tersebut
7. Menampilkan hasil identifikasi wajah tersebut pada sebuah window atau langsung pada persegi (*rectangle*)

Proses penting yang perlu diperjelas mekanismenya dari algoritma di atas antara lain adalah penerapan *face recognition*-nya. Saya menggunakan OpenCV sebagai *library* pokok untuk aplikasi ini. Untuk mendeteksi wajah pada suatu citra, OpenCV memiliki modul *CascadeClassifier* dan *LBPHFaceRecognizer,* kedua modul tersebut adalah inti dari aplikasi ini. **CascadeClassifier** adalah modul untuk mendeteksi wajah, modul ini memerlukan model yang sebelumnya sudah dilatih supaya bisa berfungsi, model sudah disediakan oleh OpenCV (ataupun pihak ketiga / *3rd party source*)dalam bentuk xml. Saya menggunakan *haarcascade\_frontalface\_alt2.xml* untuk aplikasi ini karena menurut saya lebih akurat ketimbang versi defaultnya. Sedangkan **LBPHFaceRecognizer** adalah modul untuk melatih/training dari data yang kita sediakan supaya dapat menghasilkan pola pengenalan wajah. Modul ini memiliki *method* untuk *train* dan *predict.* Pertama-tama *train* pola wajah; dari data yang sudah kita siapkan pada folder image sebagai inputnya, setelah selesai proses *training* maka akan menghasilkan model pemrediksi sebagai outputnya. Disimpan kemudian pada proses pengimplementasian model; citra yang masuk via webcam sebagai inputnya yang kemudian mencoba memprediksi wajah yang terdeteksi pada citra tersebut dalam bentuk nama dan tingkat akurasi sebagai outputnya. Tingkat akurasi pada LBPHFaceRecognizer ini memiliki nilai maksimum 0, semakin kecil nilai prediksinya mendekati angka 0 maka semakin tinggi *confident level-*nya, semakin besar nilainya maka akan semakin sedikit *confident level*-nya. Nilai prediksi ini bisa >100.

*Note:* Ada yang terlewat pada demo, Tekan “Q” pada keyboard untuk menutup aplikasi dan tekan “P” untuk menyimpan foto yang telah ter-*crop* ke dalam folder dengan *indexed naming* sesuai label.

**DAFTAR PUSTAKA**

[*https://opencv.org/about.html*](https://opencv.org/about.html)

[*https://docs.opencv.org/3.4/d7/d8b/tutorial\_py\_face\_detection.html*](https://docs.opencv.org/3.4/d7/d8b/tutorial_py_face_detection.html)

[*https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b*](https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b)