Documentazione del progetto per l’Esame di Python Programming - UFS02

Ferrarini Marco
Arificial Intelligence & Machine Learning Specialist 2024-2026
Aprile 2025

Sistema di Analisi e riconoscimento di Impronte Digitali

Panoramica del Sistema

Questo sistema è stato progettato per l'analisi, l'archiviazione e il confronto di impronte digitali. Il programma consente agli utenti di caricare immagini di impronte digitali, estrarre caratteristiche significative (minutiae e mappe di orientamento) e cercare corrispondenze tra le impronte precedentemente caricate nel database.

Il software è composto da quattro moduli principali:

- **core.py**: Contiene la logica principale per l'elaborazione delle impronte digitali
- **storage.py**: Gestisce il salvataggio e il recupero dei dati
- **ui.py**: Fornisce un'interfaccia utente a riga di comando
- **main.py**: Avvia il programma

Architettura del Sistema

Struttura delle Directory

Il sistema crea e utilizza la seguente struttura di directory:

```
Unset
|- orientationmaps/   # Mappe di orientamento salvate (.npy)
|- minutiae/          # File CSV contenenti le minutiae
|- cron_upload/       # Immagini delle impronte elaborate
(.bmp)
|- cron_ricerche/     # Risultati delle ricerche (organizzati
per data/ora)
```

Diagramma del Flusso di Lavoro

1. **Upload delle Impronte**:
    - Caricamento dell'immagine
    - Pre-elaborazione (ridimensionamento, pulizia, scheletrizzazione)
    - Estrazione delle caratteristiche (minutiae, mappa di orientamento)
    - Archiviazione dei dati
    -
2. **Ricerca di Corrispondenze**:
    - Caricamento dell'impronta di riferimento
    - Pre-elaborazione dell'impronta
    - Confronto con le mappe di orientamento nel database
    - Selezione dei 5 migliori candidati
    - Estrazione e confronto delle minutiae
    - Calcolo del punteggio finale
    - Visualizzazione e archiviazione dei risultati

Moduli del Sistema

core.py

Questo modulo contiene le classi e le funzioni principali per l'elaborazione delle impronte digitali.

Classe `fingerprint`

Rappresenta un'impronta digitale e fornisce metodi per la sua elaborazione.

Attributi:

- `id`: Identificatore univoco UUID
- `path`: Percorso del file immagine
- `resized`: Immagine ridimensionata
- `img_blur`: Immagine dopo l'applicazione del filtro bilaterale
- `img_morf`: Immagine dopo la binarizzazione e l'operazione morfologica
- `img_skel`: Immagine scheletrizzata
- `polished`: Immagine finale elaborata
- `minutiae`: DataFrame pandas contenente le minutiae
- `orientation_map`: Mappa di orientamento
- `steps_img`: Visualizzazione dei passaggi di elaborazione

Metodi:

- `rotate_fp(degrees)`: Ruota l'immagine dell'impronta

- `resize()`: Ridimensiona l'immagine a 241x298 pixel
- `polish()`: Esegue la pulizia dell'immagine e la scheletrizzazione
- `find_minutiae()`: Estrae i punti di minutiae (terminazioni e biforcazioni)
- `gen_orientation_map()`: Genera la mappa di orientamento
- `lay()`: Esegue tutti i passaggi necessari per preparare l'impronta all'analisi
- `visual_steps()`: Visualizza i passaggi di elaborazione dell'immagine

Classe `search`

Gestisce la ricerca di corrispondenze tra un'impronta di riferimento e il database.

Attributi:

- `filepath`: Percorso del file immagine
- `fingerprint_object`: Oggetto della classe fingerprint
- `minutiae`: DataFrame pandas con le minutiae
- `max_degrees`: Gradi massimi di rotazione per il confronto
- `step`: Incremento di rotazione per il confronto
- `top5maps`: 5 mappe di orientamento più simili
- `top5minutiae`: Minutiae delle 5 impronte più simili
- `matches`: Lista delle corrispondenze con punteggi
- `match_img`: Visualizzazione del confronto

Metodi:

- `find_similar_maps(maxdegrees, step)`: Trova le mappe di orientamento più simili
- `rotate_extract()`: Estrae le minutiae dalle 5 impronte più simili
- `best_match_minutiae()`: Calcola il miglior match basato su mappe di orientamento e minutiae
- `visual_match()`: Visualizza il confronto tra l'impronta di riferimento e la migliore corrispondenza

Funzioni Ausiliarie

- `plot_minutiae(img, minutiae)`: Visualizza le minutiae su un'immagine
- `compute_save_dir(dirpath)`: Elabora e salva tutte le impronte in una directory
- `compute_save(filepath)`: Elabora e salva un'impronta singola
- `rotate(map, degrees)`: Ruota un'immagine
- `save(fp, find)`: Salva i risultati di una ricerca
- `check_db()`: Verifica la presenza di impronte nel database

storage.py

Questo modulo gestisce il salvataggio e il recupero dei dati.

Funzioni

- `upload(fp)`: Salva i dati di un'impronta (mappa di orientamento, minutiae, immagine ridimensionata)
- `save_search(fp, src)`: Salva i risultati di una ricerca
- `check_db()`: Verifica la presenza di impronte nel database

ui.py

Questo modulo fornisce un'interfaccia utente a riga di comando.

Funzioni

- `main_menu()`: Menu principale che permette di scegliere tra ricerca e upload
- `retrieving_path()`: Raccoglie il percorso dell'impronta per la ricerca
- `setting_parameters()`: Permette di impostare i parametri per la ricerca
- `search(path, gradi, step)`: Esegue la ricerca
- `visualize(wanted, find)`: Visualizza i grafici
- `retrieving_path_upload()`: Raccoglie il percorso per l'upload
- `upload(percorso)`: Esegue l'upload
- `clear_screen()`: Pulisce lo schermo

Algoritmi Chiave

Pre-elaborazione delle Impronte

1. **Ridimensionamento**: L'immagine viene ridimensionata a 241x298 pixel.
2. **Normalizzazione**: I valori dei pixel vengono normalizzati nell'intervallo 0-255.
3. **Filtro Bilaterale**: Applicato per ridurre il rumore preservando i bordi.
4. **Binarizzazione Adattiva**: Converte l'immagine in bianco e nero usando una soglia adattiva gaussiana.
5. **Scheletrizzazione**: Riduce lo spessore delle creste a un pixel utilizzando la funzione `thin()` di scikit-image.

Estrazione delle Minutiae

Le minutiae sono caratteristiche distintive delle impronte digitali:

- **Terminazioni**: Punti dove una cresta finisce (un solo vicino)
- **Biforcazioni**: Punti dove una cresta si divide in due (tre o più vicini)

L'algoritmo scansiona l'immagine scheletrizzata pixel per pixel, analizzando i vicini di ogni punto per identificare queste caratteristiche.

Generazione della Mappa di Orientamento

1. Calcolo del gradiente dell'immagine usando l'operatore Sobel

2. Calcolo dei momenti del secondo ordine del gradiente
3. Applicazione di un filtro gaussiano per lisciare i momenti
4. Calcolo dell'orientazione dominante locale in ogni punto

Algoritmo di Corrispondenza

Il sistema utilizza un approccio in due fasi:

1. **Confronto delle Mappe di Orientamento**:
    - Calcolo della differenza media tra le mappe di orientamento
    - Rotazione delle mappe per trovare la migliore corrispondenza
    - Selezione delle 5 impronte più simili
2. **Confronto delle Minutiae**:
    - Calcolo della distanza euclidea tra i punti di minutiae
    - Assegnazione di pesi maggiori alle corrispondenze dello stesso tipo (terminazione o biforcazione)
    - Combinazione dei punteggi della mappa di orientamento e delle minutiae per un punteggio finale

Guida Utente

Requisiti di Sistema

- Python 3.8 o superiore
- Librerie: OpenCV, NumPy, Matplotlib, Pandas, scikit-image

Installazione

1. Assicurarsi che Python e le librerie richieste siano installati
2. Copiare i file `core.py`, `storage.py`, `ui.py` e `main.py` nella stessa directory

Utilizzo

1. Eseguire `python main.py` per avviare il programma
2. Dal menu principale, scegliere:
    - Opzione 1: Cercare una corrispondenza nel database
    - Opzione 2: Caricare nuove impronte nel database
    - 'q': Uscire dal programma

Caricamento delle Impronte

1. Selezionare l'opzione 2 dal menu principale
2. Inserire il percorso di un'immagine o di una directory contenente immagini
3. Il sistema elaborerà e salverà le impronte nel database

Ricerca di Corrispondenze

1. Selezionare l'opzione 1 dal menu principale
2. Inserire il percorso dell'immagine dell'impronta da cercare
3. Scegliere se modificare i parametri di rotazione predefiniti
4. Attendere il completamento dell'analisi
5. Visualizzare i risultati e scegliere se salvare la ricerca

Formati di Immagine Supportati

- JPG/JPEG
- PNG
- GIF
- BMP
- TIFF

Considerazioni Tecniche

Ottimizzazione delle Prestazioni

- La ricerca può richiedere tempo significativo con molte impronte nel database
- Parametri di rotazione elevati (max_degrees) e passi piccoli (step) aumentano il tempo di elaborazione
- Si consiglia un rapporto max_degrees:step intorno a 15:1 per un buon equilibrio

Limitazioni

- La qualità dell'immagine influisce significativamente sulla precisione dell'analisi
- Il sistema non è ottimizzato per impronte parziali o gravemente danneggiate
- Non vengono utilizzate tecniche di deep learning per l'estrazione delle caratteristiche

Appendice

Formato dei Dati Salvati

- **Mappe di Orientamento**: File NumPy (.npy) contenenti matrici di angoli in gradi
- **Minutiae**: File CSV con colonne "x", "y", "tipo"
- **Immagini**: File BMP contenenti le impronte elaborate
- **Risultati di Ricerca**: Directory contenenti l'immagine originale, i risultati in formato CSV e le visualizzazioni degli eventuali grafici