Projekt - Komunikacja Człowiek - Komputer

Rozpoznawanie logotypów marek laptopów

Maciej Walczykowski 145389
Szczepan Mierzejewski 140748

1 Wstęp

Projekt polega na implementacji rozwiązania by móc rozpoznawać loga wybranych 6 marek laptopów oraz ocenienie poprwności działania programu.

- apple
- asus
- dell
- hp
- huawei
- microsoft surface

Podczas implementacji korzystaliśmy z języka Python z wykorzystaniem bibliotek:

- numpy
- pylab
- cv2
- skimage
- matplotlib
- PIL

2 Plan rozwiązania

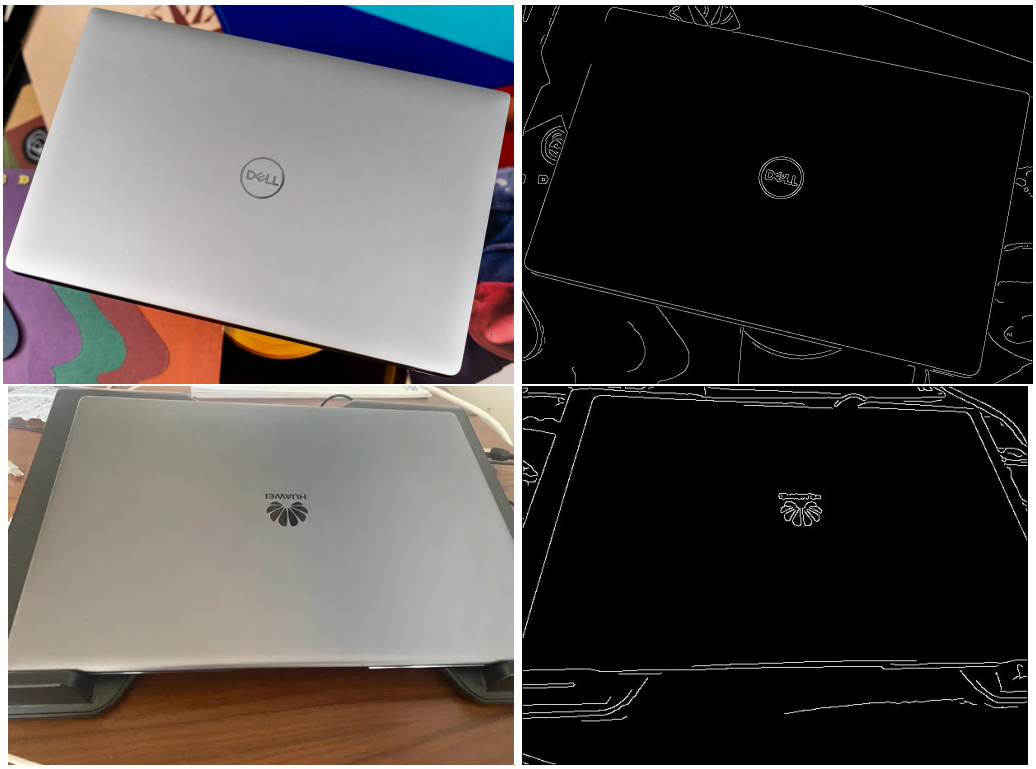
W celu odróżniania i przypisywania logotypów do marek uwzględnionych w zadaniu wykorzystane zostały momenty Hu z dodatkowymi deskryptorami wyszukujące odpowiednie kształty w celu zawężenia rozważań odnośnie badanego logotypu przez program.

Dalej, na podstawie wyliczonych momentów program powinien dla kolejnych zdjęć definiować, z jak największą dokładnością, markę danego urządzenia.

3 Przygotowanie danych

Aby algorytm miał dane bazowe, w odniesieniu do których będzie oceniał następne zdjęcia, zebraliśmy po 25 zdjęć dla każdej marki którą uwzględniliśmy w zadaniu. Zdjęcia były dobierane tak, by różniły się pozycją loga na zdjęciu, obrotem oraz oświetleniem i teksturą.

Następnie zdjęcia zostały zmodyfikowane za pomocą filtru Canny'ego. Otrzymane kontury posłużą jako bazowe dane dla momentów Hu.



Przykłady działania filtru Canny'ego

4 Rozpoznawanie logotypów

4.1 Momenty Hu

Na podstawie konturów zdjęć obliczane są momenty Hu służące do późniejszego rozpoznawania logotypów.

Do obliczenia momentów Hu najpierw korzysta się z następującego wzoru

$$M_{ij} = \sum_x \sum_y x^i y^j I(x, y)$$

Wzór na surowy moment obrazu

Obliczone w ten sposób momenty, nazywane surowymi, wykorzystywane są do wyliczenia momentów centralnych obrazu.

$$\mu_{ij} = \sum_x \sum_y (x - \bar{x})^i (y - \bar{y})^j I(x, y)$$

Wzór na centralny moment obrazu

Gdzie

$$\bar{x} = \frac{m_{10}}{m_{00}} \qquad \bar{y} = \frac{m_{01}}{m_{00}}$$

Wzory na centroidy obrazu

Następnie algorytm normalizuje momenty centralne aby uodpornić je na skalowanie.

$$\eta_{ij} = \frac{\mu_{i,j}}{\mu_{00}^{(i+j)/2+1}}$$

Wzór na znormalizowany moment centralny

Znormalizowane momenty są wykorzystywane do wyliczania momentów Hu.

$$h_0 = \eta_{20} + \eta_{02}$$
$$h_1 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$
$$h_2 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$
$$h_3 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$
$$h_4 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$
$$h_5 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2 + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})]$$
$$h_6 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

Wzory na momenty Hu

Momenty Hu charakteryzują się odpornością na transformacje takie jak obrót, czy wcześniej wspomniane skalowanie obrazu.

Jak można wywnioskować z powyższych wzorów, momentów Hu jest 7 (od 0 do 6)

Do wyliczenia owych momentów w implementacji wykorzystaliśmy bibliotekę openCV do pythona.

Po wyliczeniu wszystkich momentów dla każdego zdjęcia, program posiada bazę danych, do której można porównywać zdjęcia.

4.2 kNN

Algorytm kNN jest algorytmem wyszukującym najbliższego sąsiada dla podanej wartości(bądź zbioru wartości)
(ang.: k-nearest neighbors)

Działanie algorytmu:

1. Ładowanie danych
2. Inicjalizacja k dla wybranej liczby sąsiadów
3. Dla każdego przykładu:
   (a) Obliczenie odległości między przykładem a obiektem zawartym w zapytaniu.
   (b) Dodanie odległości wraz z indeksem do uporządkowanego zbioru.
4. Uporządkowanie pełnego zbioru rosnąco względem obliczonej odległości.
5. Wybranie pierwszych k wpisów ze zbioru.
6. Sprawdzenie etykiet wybranych wpisów k.
7. W zależności od wyniku:
   - Regresja - Zwrócenie średniej ze sprawdzonych etykiet.
   - Klasyfikacja - Zwrócenie dominanty ze sprawdzonych etykiet.

Podczas implementcji algorytmu kNN należy pamiętać o dobraniu opowiedniego k. Nie może być ono zbyt małe, w szczególności nie powinno być równe 1. W takim przypadku może się okazać, że prosty do wyeliminowania błąd będzie obecny w rozwiązaniu.

Jeśli k będzie wyjątkowo duże, ucierpi na tym prędkość wykonywania się programu. kNN, mimo tego, jest prostym algorytmem do zaimplementowania.

4.3 Rozpoznawanie zdjęć testowych

W ramach testowania rozwiązania przygotowaliśmy zdjęcia, które nie pokrywają się ze zdjęciami na podstwie których wyznaczamy momenty Hu.

	apple	asus	dell	hp	huawei	microsoft
apple	73.53%	2.94%	5.88%	2.94%	14.71%	0%
asus	5.88%	76.47%	5.88%	2.94%	8.82%	0%
dell	0%	0%	82.35%	2.94%	5.88%	8.82%
hp	11.76%	2.94%	2.94%	70.59%	8.82%	2.94%
huawei	8.82%	0%	5.88%	5.88%	73.53%	5.88%
microsoft	2.94%	2.94%	5.88%	5.88%	5.88%	76.47%

Macierz pomyłek dla wykonanych testów

Jak widać algorytm w każdym przypadku większość logotypów rozpoznaje poprawnie. Najmniejsza wartość pomyślnych klasyfikacji wynosi 70.59%, występuje ona dla marki *hp*. Natomiast najwyższa wartość należy do marki *dell*, wynosi ona 82.35%.

Wyniki, choć nienajniższe, mogłyby być lepsze. Prawdopodobnie udoskonalenie preprocessingu, w programie wyeliminowałoby część fałszywych klasyfikacji.

5 Literatura

1. Analysis of Hu's moment invariants on image scaling and rotation
2. Machine Learning Basics with the K-Nearest Neighbors Algorithm