JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 10 (2019) 67-70
Wysłane: 2018-12-02
Przyjęte: 2018-12-07

Porównanie skuteczności wybranych algorytmów rozpoznawania twarzy w przypadku zdjęć o niskiej jakości

Jakub Gozdur*, Bartosz Wiśniewski, Piotr Kopniak
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Celem artykułu jest określenie skuteczności popularnych algorytmów rozpoznawania twarzy w przypadku zdjęć o niskiej jakości. W trakcie pracy zostały opisane podstawowe algorytmy rozpoznawania twarzy takie jak LBPH, Eigenfaces i Fisherfaces. Do przeprowadzenia badań stworzono platformę badawczą wyposażona w oprogramowanie pozwalające testować dane i zbierać wyniki. Rezultaty badań pokazała, że jedynym algorytmem nadającym się do takich rozwiązań jest LBPH. Pozostałe natomiast nie uzyskały odpowiednio wysokiego współczynnika skuteczności.

Słowa kluczowe: rozpoznawanie; twarz; LBPH; Eigenfaces; Fisherfaces

*Autor do korespondencji.

Adres e-mail: jakub.gozdur@pollub.edu.pl

Comparison of the effectiveness of selected face recognition algorithms for poor quality photos

Jakub Gozdur*, Bartosz Wiśniewski, Piotr Kopniak
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The goal of the article is to determine the effectiveness of popular face recognition algorithms for poor quality photos. Basic facial recognition algorithms such as LBPH, Eigenfaces and Fisherfaces were described during the work. A research platform equipped with software allowing to test data and collect results was created. The results of the research showed that the only algorithm suitable for such solutions is LBPH. The others, however, did not achieve a sufficiently high effectiveness factor.

Keywords: recognition; face; LBPH

*Corresponding author.

E-mail addresses: jakub.gozdur@pollub.edu.pl

1. Wstęp

W dzisiejszych czasach technologie rozpoznawania twarzy są coraz częściej wykorzystywane. Najlepszym przykładem na to są telefony komórkowe lub inteligentne domy z systemami zabezpieczeń opartymi na detekcji twarzy. Mimo to, że te technologie i narzędzia osiągnęły wysoki poziom to w dalszym ciągu nie są one doskonałe i wymagają dalszych badań.

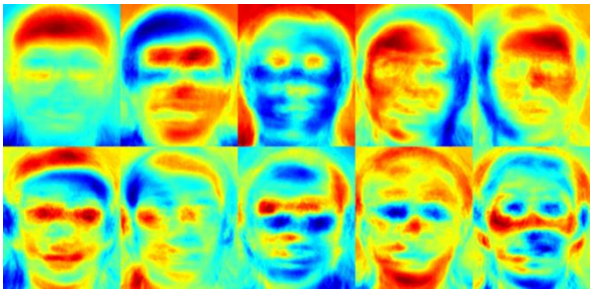
2. Metody rozpoznawania twarzy

Kierując się popularnością używanych rozwiązań wybrane zostały trzy algorytmy rozpoznawania twarzy, które jednocześnie odznaczają się stosunkową łatwością w implementacji:

- Eigenfaces;
- Fisherfaces;
- LBPH.

2.1. Eigenfaces

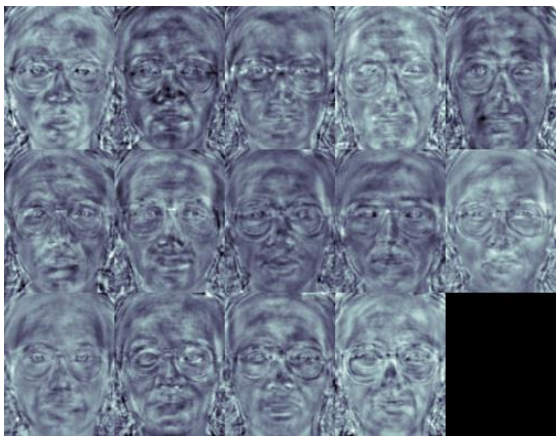
Eigenfaces bierze pod uwagę fakt, że nie wszystkie części twarzy są równie ważne lub równie przydatne przy rozpoznawaniu twarzy. Ma to odzwierciedlenie w rzeczywistości, ponieważ gdy patrzymy na kogoś, możemy tę osobę rozpoznać poprzez jej cechy szczególne, takie jak oczy, nos, policzki lub czoło; i tym jak bardzo różnią się one względem siebie. Dokładniej, koncentrujemy się na obszarach najbardziej różniących się od siebie. Na przykład, różnica pomiędzy obszarem oczu, a obszarem nosa jest zdecydowanie zauważalna. Gdy patrzymy na wiele twarzy, porównujemy je, patrząc właśnie na te regiony, ponieważ dzięki uchwyceniu maksymalnych różnic między twarzami możemy odróżnić jedną od drugiej. W ten właśnie sposób działa funkcja rozpoznawania Eigenfaces. Analizuje wszystkie obrazy treningowe wszystkich osób jako całości i próbuje wydobyć składniki, które są istotne i użyteczne, a odrzuca pozostałe. Te najbardziej znaczące cechy nazywane są głównymi komponentami [2, 4]. Rysunek 1 przedstawia cechy wyodrębnione z twarzy. Można zauważyć, że cechy charakterystyczne reprezentujące twarze są zaznaczone odpowiednimi kolorami. Stąd właśnie algorytm Eigenfaces otrzymał swoją nazwę: z jęz. niem. Eigen - swój/własny.



Rys. 1. Algorytm Eigenfaces – chechy charakterystyczne twarzy [5]

2.2. Fisherfaces

Fisherfaces uważany jest jako ulepszona wersja Eigenfaces. Poprzedni algorytm analizuje wszystkie twarze treningowe naraz i wyszukuje główne komponenty wszystkich z nich. W ten sposób nie koncentruje się na cechach, które jednoznacznie wykluczają podobieństwo jednej osoby do drugiej. Ponieważ Eigenfaces uwzględnia oświetlenie jako istotne kryterium oceny, zakwalifikuje tę zmiany jako użyteczne dla rozpoznawania twarzy, w wyniku czego może odrzucić cechy twarzy innych osób, uznając je za mało przydatne, prowadząc do błędnej klasyfikacji. Co prawda można tego uniknąć, konfigurując algorytm Eigenfaces, tak aby wyodrębnił użyteczne cechy z twarzy każdej osoby oddzielnie, zamiast ze wszystkich naraz. W ten sposób, nawet jeśli dane jednej osoby mają duże zmiany w oświetleniu, nie wpłynie to na proces wyodrębniania cech innych osób. Natomiast, algorytm rozpoznawania twarzy Fisherfaces wyodrębnia główne komponenty, które odróżniają jedną osobę od innych. W tym przypadku komponenty jednostki nie dominują (stają się bardziej użyteczne) nad innymi. Poniżej znajduje się obraz głównych komponentów wyodrębnionych za pomocą algorytmu Fisherfaces.

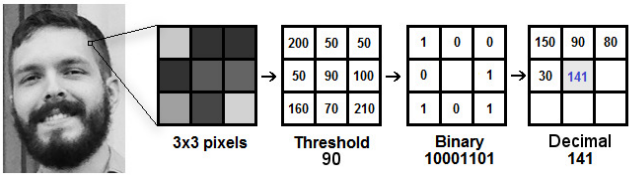


Rys.2. Algorytm Fisherfaces – cechy charatreystczne twarzy [5]

Należy zwrócić uwagę na to, że moduł FisherFaces zapobiega tylko dominacji jednej osoby, lecz jego wadą jest to, że analizując twarz brane pod uwagę jest jej oświetlenie. Wiemy natomiast, że zmienność światła nie jest użyteczną funkcją do wyodrębnienia, ponieważ nie jest częścią rzeczywistej twarzy [1, 4].

2.3. LBPH

Główna idea LBPH nie polega na oglądaniu obrazu jako całości, ale na próbie znalezienia lokalnej struktury przez porównanie każdego piksela z sąsiednimi pikselami. Proces rozpoznawania twarzy algorytmu LBPH można zademonstrować rysując okno 3 × 3 i przesuwając je po obrazie. Przy każdym przesunięciu należy porównać wartość piksela w środku z wartościami pikseli otaczających go. W przypadku gdy wartość sąsiada jest mniejsza lub równa centralnemu, przypisujemy mu 0, a w przeciwnym wypadku 1 [3]. Po posortowaniu etykiet pikseli w oknie 3 × 3 w kolejności zgodnej z ruchem wskazówek zegara, otrzymujemy wzór binarny, który jest lokalny dla określonego obszaru obrazu. Po zakończeniu procesu dla całego obrazu, otrzymujemy listę lokalnych wzorów binarnych.



Rys. 3. Local Binary Pattern Histograms Fisherfa – zasada działania [5]

Następnie, po uzyskaniu listy lokalnych wzorców binarnych, konwertuje się każdy z nich na liczbę dziesiętną a następnie tworzy się histogram wszystkich wartości dziesiętnych [4].

W celu dokładniejszego zapoznania się z zasadami działania algorytmów, odsyłamy do pozycji literaturowych znajdującymi się w bibliografii.

3. Cel i plan badań

Celem badań jest określenie skuteczności wybranych algorytmów rozpoznawania twarzy w przypadku zdjęć o niskiej jakości. Temat ten został podjęty, aby zbadać efektywność rozpoznawania twarzy w życiu codziennym gdzie kamera nie zawsze jest wysokiej jakości oraz kiedy nie można zagwarantować idealnych warunków atmosferycznych. Należy potwierdzić lub odrzucić następującą hipotezę:

Algorytmy LBPH, Fisherfaces, Eigenfaces mogą być wykorzystywane do rozpoznawania twarzy w przypadku zdjęć o niskiej jakości.

Przez zdjęcia o niskiej jakości rozumie się fotografię o niskiej rozdzielczości lub rozmytym obrazie. Do przeprowadzenia badań została stworzona dedykowana platforma badawcza wykorzystująca minikomputer Raspberry Pi Zero W wyposażoną w dedykowaną kamerę o rozdzielczości 8 Mpx. Ponadto, zostało zaimplementowane oprogramowanie wykorzystujące bibliotekę OpenCv, która umożliwia wykorzystanie ww. algorytmów.

Przykład 1. Kod programu

```
#include <stdio.h>
import cv2
import os
import numpy as np
subjects = ["", "wisnia", "kubus", "kamila", "mohi", "nati"]
def detect_face(img):
```

```
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    face_cascade = cv2.CascadeClassifier('opencv-
files/lbpcascade_frontalface.xml')
    faces = face_cascade.detectMultiScale(gray,
scaleFactor=1.3, minNeighbors=4);
    if (len(faces) == 0):
        return None, None
    (x, y, w, h) = faces[0]
    return gray[y:y+w, x:x+h], faces[0]
def prepare_training_data(data_folder_path):
    dirs = os.listdir(data_folder_path)
    faces = []
    labels = []
    for dir_name in dirs:
        if not dir_name.startswith("s"):
            continue;
        label = int(dir_name.replace("s", ""))
        subject_dir_path = data_folder_path + "/" + dir_name
        subject_images_names = os.listdir(subject_dir_path)
        for image_name in subject_images_names:
            if image_name.startswith("."):
                continue;
            image_path = subject_dir_path + "/" + image_name
            image = cv2.imread(image_path)
            cv2.imshow("Training on image...", cv2.resize(image,
(400, 500)))
            cv2.waitKey(100)
            face, rect = detect_face(image)
            if face is not None:
                faces.append(face)
                labels.append(label)
    cv2.destroyAllWindows()
    cv2.waitKey(1)
    cv2.destroyAllWindows()
    return faces, labels
print("Preparing data...")
faces, labels = prepare_training_data("training-data")
print("Data prepared")
print("Total faces: ", len(faces))
print("Total labels: ", len(labels))
face_recognizer = cv2.face.HaarFaceRecognizer_create()
face_recognizer.train(faces, np.array(labels))
def draw_rectangle(img, rect):
    (x, y, w, h) = rect
    cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 2)
def draw_text(img, text, x, y):
    cv2.putText(img, text, (x, y), cv2.FONT_HERSHEY_PLAIN,
1.5, (0, 255, 0), 2)
def predict(test_img):
    img = test_img.copy()
    face, rect = detect_face(img)
    label, confidence = face_recognizer.predict(face)
    label_text = subjects[label]
    draw_rectangle(img, rect)
    draw_text(img, label_text, rect[0], rect[1]-5)
    return img
print("Predicting images...")
test_img = cv2.imread("test-data/test1.jpg")
predicted_img = predict(test_img)
cv2.imshow(cv2.resize(predicted_img, (400, 500)))
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Napisany program odpowiedzialny jest za przekształcenie zdjęć szkoleniowych w odpowiednie dane treningowe wykorzystując funkcje prepare_treining_data(PATH) oraz detect_face(IMG), która realizuje detekcje twarzy. Kolejno algorytm jest inicjalizowany oraz uczony wykorzystując dane treningowe. Ostatnim a zarazem najważniejszym zadaniem programu jest samo rozpoznanie twarzy na dostarczonym zdjęciu. Jest to możliwe dzięki funkcji predict(IMG). Dodatkowo dołączone są dwie funkcji odpowiedziane za formatowanie zdjęcia wynikowego - draw_rectangle(IMG, RECTANGLE), draw_text(IMG, TEXT, X, Y). Stworzony program realizuje funkcjonalności rzeczywistych systemów rozpoznawania oraz może być użyty do symulacji sytuacji z życia codziennego. Dodatkowo może być z łatwością powielony i rozwijany wedle potrzeb.

W celu określenia skuteczności algorytmów przeprowadzono szereg badań. Zostały one przeprowadzone z udziałem 5 osób dla których wykonano po 10 zdjęć badawczych wykorzystując stworzoną platformę. Do wykrycia obszaru twarzy został użyty wbudowany klasyfikator Haar biblioteki OpenCV przeszkolony na tysiącach twarzy. Zdjęcia badawcze zostały zrobione obniżając rozdzielczość oraz przysłaniając obiektyw kamery.

Rys. 4. Zdjęcia o niskiej jakości

Wynikami badań są trzy wartości:

- Skuteczność – procentowa wartość pozytywnego dopasowania osoby do zdjęcia;
- Czas predykcji;
- Pewność – wartość określająca odległość do najbliższego elementu w bazie znanych twarzy, mniejsza wartość oznacza większą pewność.

4. Wyniki badań

Pierwszym algorytmem poddanym badaniom był algorytm LBPH. Uzyskane wyniki przedstawia tabela 1

Tabela 1. Uśrednione wyniki rozpoznawania twarzy z wykorzystaniem algorytmu LBPH

Skuteczność	Czas predykcji [s]	Pewność
78%	4,16	76.059

Za zadowalającą skuteczność można uznać wynik w okolicy 80%. Badania pokazują, że LBPH znalazł się na granicy tej wartości. Porównując uzyskany wynik z otrzymanymi wartościami efektywności dla zdjęć o dobrej jakości, uzyskanymi w efekcie analogicznych badań, która wynosi w okoli 92% można uznać, że LBPH zadowalająco radzi sobie z zdjęciami o niskiej jakości. Średni czas wyniósł 4,16 sekund co w przypadku użytego sprzętu jest zadowalającym wynikiem. Docelowa pewność wyznaczona na podstawie

własnych badań na ponad 300 zdjęciach dla tego algorytmu wynosi w okolicach 20. Dużo wyższa wartość uzyskana w badaniach świadczy o tym, że rozpoznanie straciło na jakości.

Drugim algorytmem poddanym badaniom był Fisherfaces, uzyskane wyniki przedstawia tabela 2.

Tabela 2. Uśrednione wyniki rozpoznawania twarzy z wykorzystaniem algorytmu Fisherfaces

Skuteczność	Czas predykcji [s]	Pewność
44%	3,93	3126

Uzyskana skuteczność jest wynikiem nie zadowalającym co oznacza, że algorytm nie może być skutecznie użyty w systemach rozpoznania twarzy obsługującym zdjęcia o niskiej jakości.

Ostatnim przetestowanym algorytmem był Eigenfaces, którego wyniki przedstawia tabela 3.

Tabela 3. Uśrednione wyniki rozpoznawania twarzy z wykorzystaniem algorytmu Eigenfaces

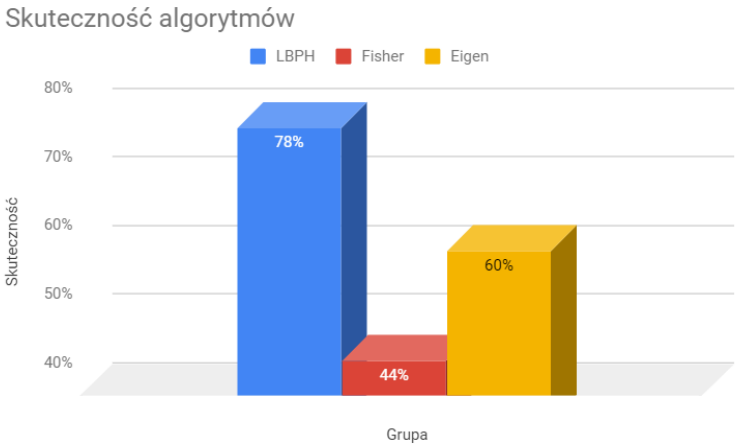
Skuteczność	Czas predykcji [s]	Pewność
60%	4,44	8894

Osiągnięta skuteczność jest również nie zadowalającym wynikiem, co świadczy o tym, że algorytm nie powinien być wykorzystywany ww. scenariuszach.

5. Wnioski

Podstawioną hipotezę, że **Algorytmy LBPH, Fisherfaces, Eigenfaces mogą być wykorzystywane do rozpoznawania twarzy w przypadku zdjęć o niskiej jakości** udało się udowodnić tylko w przypadku algorytmu LBPH. Pozostałe algorytmy nie osiągnęły zadowalających wyników.

Czas predykcji we wszystkich przypadkach okazał się być zbliżony, więc w przypadku rozwiązań gdzie szybkość działania nie jest najwyższym priorytetem nie powinna być brana pod uwagę jako czynnik decydujący. Warto zauważyć, iż mimo to, że algorytm LBPH osiągnął zadowalającą skuteczność, niekorzystna wartość pewności może świadczyć o nieprzewidywalności algorytmu z tego względu, więc nie powinien być on wykorzystywany w rozwiązaniach gdzie dominującym założeniem jest niedopuszczenie fałszywego rozpoznania. Poniższy wykres przedstawia zestawienie skuteczności algorytmów.



Rys.5. Skuteczność algorytmów

Literatura

[1] Stan Z. Li, Anil K. Jain: Handbook of Face Recognition. Springer. New York 2011.

[2] Ahonen, T., Hadid, A., and Pietikainen, M.: Face Recognition with Local Binary Patterns. Springer. Heidelberg 2004.

[3] Asit K. Datta, Madhura Datta, Pradipta K. Banerjee: Face Detection and Recognition: Theory and Practice. CRC Press. Boca Raton 2015

[4] Peter N. Belhumeur, Joao P. Hespanha, and David J. Kriegman: Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection. IEEE Transactions on Pattern Analysis and Machine Intelligence. Tom 19 Nr 7/1997

[5] Super data science, https://www.superdatascience.com/opencv-face-recognition/ [15.05.2018].