УДК 004

Применение алгоритмов искусственного интеллекта для распознавания лиц с изображений

Вихляев Дмитрий Романович
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье рассматриваются алгоритм и практическое описание программы для распознавания лиц с изображений. Программа разработана с помощью языка программирования python и библиотек обработки изображений и машинного обучения. Результатом исследования станет подробное описание программы распознавания лиц человека.
Ключевые слова: распознавание изображений, нейронные сети, face-recognition.

Application of artificial intelligence algorithms for face recognition from images

Vikhlyaev Dmitry Romanovich
Sholom-Aleichem Priamursky State University
Student

Abstract
This article discusses the algorithm and the practical description of the program for face recognition in images. The program is developed using the python programming language and image processing and machine learning libraries. The result of the study will be a detailed description of the human face recognition program
Keywords: image recognition, neural networks, face-recognition.

1 Введение

1.1 Актуальность

Распознавание лиц приобрело свою популярность вначале 2000-х, когда Пол Виола и Майкл Джонс изобрели способ обнаружения лиц, который был достаточно быстрым, чтобы работать на дешевых камерах. Хотя, сейчас существуют гораздо более надежные решения данной проблемы. Функция распознавания лиц по сей день, внедрена в камеры, благодаря чему, она может убедиться, что все лица находятся в фокусе, прежде чем сделать снимок. Но распознавание лиц на самом деле представляет собой ряд связанных между собой проблем, из-за которых даже современные нейронные сети не могут, всегда определить расположение и распознавание лица. Сначала на картинке необходимо найти все лица. Человек способен узнавать лицо даже у неживых

предметов, например, в скалах или у автомобиля с фронтальной стороны. Для машины данная задача становится невообразимо сложной так как изображение может иметь слабое освещение или невыраженные границы. Благодаря огромному количеству препятствий исследование в сфере компьютерного зрения по сей день, являются актуальными.

1.2Обзор исследований

В.М.Дуденков реализовал распознавание изображений с помощью самоорганизующихся карт кохонена и сетей нечеткой логики [1]. В.Н.Ярмолюк провёл исследование узла вращения изображения в оптическом корреляторе распознавания изображений с произвольным положением и ориентацией [2]. М.П.Кривенко описал байесовский подход в распознавании фрагментов изображения, имеющих различные размеры [3]. Х.Мризаи показал различные выборы начальных условий обучения нейронных сетей с многомерными входными данными [4]. З.Т.Нгуен, В.М.Хачумов построили модели и методы сопоставления изображений в задаче распознавания лиц [5].

1.3Цель исследования

Цель исследования – разработать программу распознавания лиц человека при обработке изображений.

2 Материалы и методы

В данном исследовании используются сторонние модули, имеющие интерфейс для работы с языком программирования python. Для работы с изображениями применяются библиотеки opencv и skimage. Применение нейронных сетей осуществляется с помощью модулей dlib и face-recognition.

3 Результаты и обсуждения

Алгоритм распознавания лиц состоит из обнаружения лица на изображении, идентификация лица при искажении повороте и освещении, выделении уникальных черт лица, сравнение уникальностей всех известных лиц, чтобы определить имя человека.

Далее будет использоваться метод, изобретенный в 2005 году, под названием HOG (Гистограмма ориентированных градиентов). Для начала изображения будут представлены в черно-белом виде, потому что данные о цвете не нужны для поиска лиц. Затем будет рассмотрен каждый пиксель изображения по очереди. Для каждого отдельного пикселя будут посмотрены пиксели, непосредственно окружающие его. Целью является выяснить, насколько темным является текущий пиксель по сравнению с пикселями, непосредственно окружающими его. После чего, рисуется стрелка, показывающая, в каком направлении изображение становится темнее. Если повторить этот процесс для каждого отдельного пикселя, то каждый пиксель, заменится стрелкой. Эти стрелки называются градиентами и показывают переход от светлого оттенка к темному по всему изображению.

Изображение разбивается на маленькие квадраты размером 16x16 пикселей каждый. В каждом квадрате подсчитывается, сколько градиентов указывает в каждом основном направлении. Затем квадрат заменяется на изображении направлениями стрелок, которые были самыми сильными.

Конечным результатом является то, что исходное изображение становится представлением, которое простым способом отражает основную структуру лица.

Чтобы найти лица на этом изображении нужно лишь найти часть изображения, который был извлечен из множества других тренировочных лиц (рис.1,2).

```
from skimage.feature import hog
from skimage import exposure
import cv2
import matplotlib.pyplot as plt


image = cv2.imread('q.jpg')
fd, hog_image = hog(image, orientations=8, pixels_per_cell=(16, 16),
                    cells_per_block=(1, 1), visualize=True, multichannel=True)

fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(8, 4), sharex=True, sharey=True)

ax1.axis('off')
ax1.imshow(image, cmap=plt.cm.gray)
ax1.set_title('Input image')

# Rescale histogram for better display
hog_image_rescaled = exposure.rescale_intensity(hog_image, in_range=(0, 10))

ax2.axis('off')
ax2.imshow(hog_image_rescaled, cmap=plt.cm.gray)
ax2.set_title('Histogram of Oriented Gradients')
plt.show()
```

Рис. 1. Генерация HOG-представления изображения

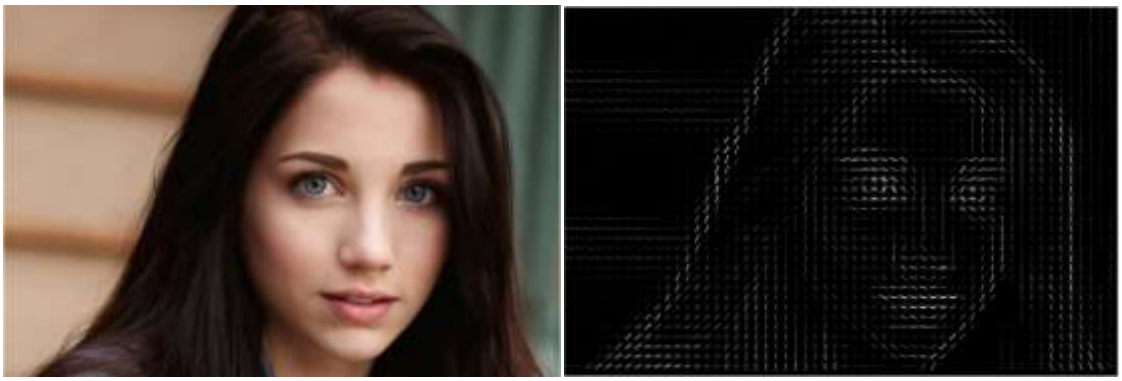

Рис. 2. Оригинальное и HOG-представление одного изображения

Теперь нужно решить проблему, заключающуюся в том, что лица, повернутые в разные стороны, выглядят для компьютера совершенно по-другому.

Чтобы учесть это, происходит, попытаемся деформировать каждое изображение так, чтобы глаза и губы всегда находились в месте образца на изображении. Это значительно облегчит сравнение лиц на следующих шагах.

Для этого будет использован алгоритм, называемый оценкой ориентира лица, изобретенным в 2014 году Вахидом Каземи и Жозефиной Салливан.

Основная идея заключается в том, что создаётся 68 определенных точек (ориентиров), которые существуют на каждом лице — верхняя часть подбородка, внешний край каждого глаза, внутренний край каждой брови и т. д. Затем запускается алгоритм обучения, чтобы уметь находить эти 68 определенных точек на любом лице.

Теперь, когда известно, где находятся глаза и рот, можно просто повернуть, масштабировать и сдвигать изображение так, чтобы глаза и рот располагались как можно лучше по центру. Для этого будут использоваться только основные преобразования изображения, такие как вращение и масштабирование, они сохраняют параллельные линии. Такой метод называется аффинным преобразованием.

Теперь независимо от того, как повернуто лицо, можно центрировать глаза и рот примерно в одном и том же положении на изображении. Это сделает следующий шаг более точным (рис.3).

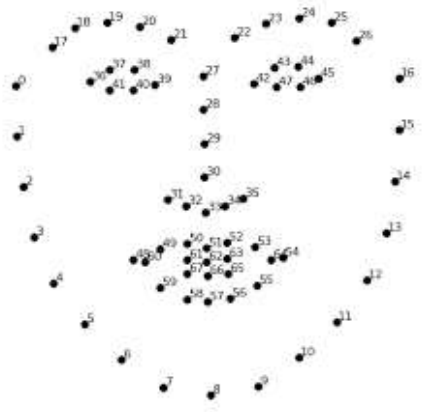

Рис. 3. Расположение ориентиров лица

Реализация данного метода представлена с использованием модуля dlib и данных содержащих информацию о точках лица для обучения модели [8]. Dlib – это современный набор инструментов C++, содержащий алгоритмы машинного обучения и инструменты для создания сложного программного обеспечения. Библиотека имеет API для взаимодействия с python (рис.4,5).

```
import sys
import dlib
from skimage import io

predictor_model = "shape_predictor_68_face_landmarks.dat"

file_name = sys.argv[1]

face_detector = dlib.get_frontal_face_detector()
face_pose_predictor = dlib.shape_predictor(predictor_model)

win = dlib.image_window()

file_name = sys.argv[1]

image = io.imread(file_name)

detected_faces = face_detector(image, 1)

print("Found {} faces in the image file {}".format(
        len(detected_faces),
        file_name))

win.set_image(image)

for i, face_rect in enumerate(detected_faces):

        print("- Face #{} found at Left: {} Top: {} Right: {} Bottom: {}".format(
        i,
        face_rect.left(),
        face_rect.top(),
        face_rect.right(),
        face_rect.bottom()))

        win.add_overlay(face_rect)

        pose_landmarks = face_pose_predictor(image, face_rect)

        win.add_overlay(pose_landmarks)

dlib.hit_enter_to_continue()
```

Рис. 4. Нахождение ориентиров лица

Рис. 5. Результат нахождение ориентиров лица

Следующим шагом является, различение лиц. Самый простой подход к распознаванию лиц – это прямое сравнение неизвестного лица, найденного на шаге 2, со всеми имеющимися фотографиями людей, которые уже были помечены. Необходим способ извлечения нескольких основных измерений с каждой грани. Затем таким же образом измерить неизвестное лицо и найти известное с самыми близкими измерениями. Например, измерить размер каждого уха, расстояние между глазами, длину носа и т. д. Решение состоит в обучении глубокой сверхточной нейронной сети генерировать 128 измерений для каждого лица.

Процесс обучения работает, просматривая 3 изображения лица одновременно. Два изображения относятся к одному человеку и одно к другому. Затем алгоритм просматривает измерения, которые в настоящее время генерирует для каждого из этих трех изображений. Затем немного настраивает нейронную сеть, чтобы убедиться, что измерения, которые она генерирует для № 1 и № 2, немного ближе, а измерения для № 2 и № 3 немного дальше друг от друга. Повторив этот шаг миллионы раз для миллионов изображений тысяч разных людей, нейронная сеть научится надежно генерировать 128 измерений для каждого человека. Любые десять разных изображений одного и того же человека должны давать примерно одинаковые размеры.

Для создания программы распознавания используются модули opencv и face-recognition. Сначала загружается изображение с помощью метода библиотеки face-recognition, затем конвертируется из формата BGR в RGB. В подготовленном изображении обнаруживается фрагмент лица через точки подчёркивания. Метод face_locations возвращает массив координат. Поскольку в данном примере в изображении содержится только одно лицо, возвращаются данные нулевого элемента. Данные обнаруженного лица кодируется для вывода на изображение. Метод cv2.rectangle обрисовывает прямоугольник по координатам обнаруженного лица.

Следующим шагом является, различение лиц. Самый простой подход к распознаванию лиц – это прямое сравнение неизвестного лица, найденного на шаге 2, со всеми имеющимися фотографиями людей, которые уже были помечены. Необходим способ извлечения нескольких основных измерений с каждой грани. Затем таким же образом измерить неизвестное лицо и найти известное с самыми близкими измерениями. Например, измерить размер каждого уха, расстояние между глазами, длину носа и т. д. Решение состоит в обучении глубокой сверхточной нейронной сети генерировать 128 измерений для каждого лица.

Сравнение лиц происходит путём сравнения кодировок между новым и загруженными лицами с помощью метода compare_faces. В качестве параметров метод принимает список закодированных данных лиц, из которых нужно найти определённое и сравниваемое лицо.

Результат распознавания отображается на картинке через метод cv2.putText (рис.6,7,8,9).

```
import cv2
import dlib
import face_recognition

imgGarry=face_recognition.load_image_file('34.jpg')
imgGarry=cv2.cvtColor(imgGarry,cv2.COLOR_BGR2RGB)

imgTest=face_recognition.load_image_file('334.jpg')
imgTest=cv2.cvtColor(imgTest,cv2.COLOR_BGR2RGB)

faceloc=face_recognition.face_locations(imgGarry)[0]
encodeGarry=face_recognition.face_encodings(imgGarry)[0]
cv2.rectangle(
    imgGarry,
    (faceloc[3],faceloc[0]),
    (faceloc[1],faceloc[2]),
    (255,0,255),2)

facelocTest=face_recognition.face_locations(imgTest)[0]
encodeTest=face_recognition.face_encodings(imgTest)[0]
cv2.rectangle(
    imgTest,
    (facelocTest[3],facelocTest[0]),
    (facelocTest[1],facelocTest[2]),
    (255,0,255),2)

results=face_recognition.compare_faces([encodeGarry], encodeTest)
facedis= face_recognition.face_distance([encodeGarry], encodeTest)
print(results,facedis)

cv2.putText(
    imgTest,
    f'{results} {round(facedis[0],2)}',
    (80,50),
    cv2.FONT_HERSHEY_COMPLEX,0.5,(0,255,0),1)

cv2_imshow(imgGarry)
cv2_imshow(imgTest)
```

Рис. 6. Распознавание лиц

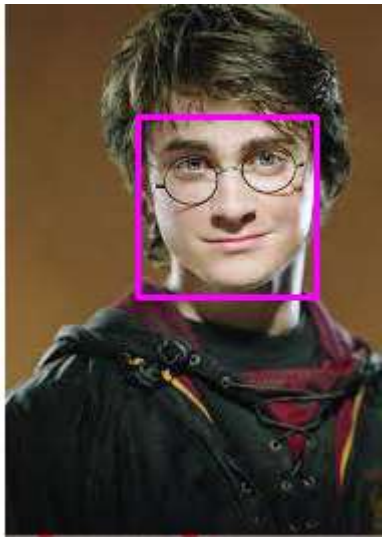

Рис. 7. Изображение для распознавания

Рис. 8. Результат программы после распознавания того же человека

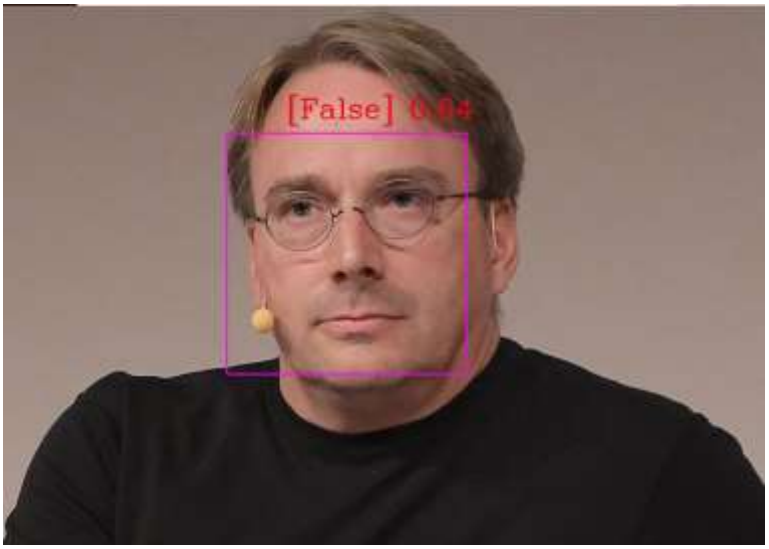

Рис. 9. Результат программы после распознавания другого человека

В результате исследования был описан алгоритм распознавания лиц и продемонстрирована работа программы с использованием библиотеки face-recognition. Полученные результаты показывают, что данная библиотека позволяет идентифицировать лица с высокой точностью с минимальными затратами.

Библиографический список

1. Дуденков В.М. Распознавание изображений с помощью самоорганизующихся карт Кохонена и сетей нечеткой логики // В сборнике: современные методы прикладной математики, теории управления и компьютерных технологий (пмтукт-2014). сборник трудов vii международной конференции. 2014. с. 131-133.

2. Ярмолюк В.Н. Исследование узла вращения изображения в оптическом корреляторе распознавания изображений с произвольными положением и ориентацией // Депонированная рукопись № 94-В2004 22.01.2004
3. Кривенко М.П. Байесовский подход в распознавании фрагментов изображения, имеющих различные размеры // Обозрение прикладной и промышленной математики. 2007. Т. 14. № 3. С. 538а-539.
4. Мризаи Х. Выбор начальных условий обучения нейронных сетей с многомерными входными данными // В сборнике: Информатика: проблемы, методология, технологии. материалы XV международной научно-методической конференции. 2015. С. 170-176.
5. Нгуен З.Т., Хачумов В.М. Модели и методы сопоставления изображений в задаче распознавания лиц // Искусственный интеллект и принятие решений. 2016. № 4. С. 5-14.
6. Гусейнов С.Б., Менгель В.В., Орозкожоев Д.С. Использование интеллектуальных систем при распознавании текста на изображении // Известия Кыргызского государственного технического университета им. И. Раззакова. 2022. № 1 (61). С. 46-52.
7. Карасев П.И. Интегральное представление изображений в распознавании образов // В сборнике: Охрана, безопасность, связь - 2014. материалы международной научно-практической конференции. Воронежский институт МВД России. 2015. С. 200-201. http://dlib.net/files/shape_predictor_68_face_landmarks.dat.bz2