Uniwersytet im. Adama Mickiewicza w Poznaniu
Wydział Matematyki i Informatyki

Informatyka
Informatyka kto wie jaka to będzie

Magdalena Mozgawa
Nr albumu: **389479**

Ataki na systemy przetwarzania obrazu

Attacks on computer vision systems

Praca magisterska

Promotor:
dr inż. Michał Ren

2021 (oby)

Poznań, dnia .....

OŚWIADCZENIE

Ja, niżej podpisany IMIONA NAZWISKO student Wydziału Matematyki i Informatyki Uniwersytetu im. Adama Mickiewicza w Poznaniu oświadczam, że przedkładaną pracę dyplomową pt: „TYTUŁ PRACY" napisałem samodzielnie. Oznacza to, że przy pisaniu pracy, poza niezbędnymi konsultacjami, nie korzystałem z pomocy innych osób, a w szczególności nie zlecałem opracowania rozprawy lub jej części innym osobom, ani nie odpisywałem tej rozprawy lub jej części od innych osób.

Oświadczam również, że egzemplarz pracy dyplomowej w wersji drukowanej jest całkowicie zgodny z egzemplarzem pracy dyplomowej w wersji elektronicznej.

Jednocześnie przyjmuję do wiadomości, że przypisanie sobie, w pracy dyplomowej, autorstwa istotnego fragmentu lub innych elementów cudzego utworu lub ustalenia naukowego stanowi podstawę stwierdzenia nieważności postępowania w sprawie nadania tytułu zawodowego.

Wyrażam zgodę na udostępnianie mojej pracy w czytelni Archiwum UAM.

Wyrażam zgodę na udostępnianie mojej pracy w zakresie koniecznym do ochrony mojego prawa do autorstwa lub praw osób trzecich.

.....

(czytelny podpis studenta)

Tutaj będą podziękowania dla sąsiadów
za siedzenie cicho.

I coś dla promotora.

Spis treści

Spis rysunków

Spis tablic

Spis kodów źródłowych

Streszczenie

Streszczenie wstępu jest dobrym pomysłem na początek abstraktu. Dobre praktyki tworzenia abstraktów znajdują się np. na stronie[1]. Wczuj się w rolę informatyka, który będzie czytał sam abstrakt, żeby zdecydować, czy reszta pracy mu się przyda. Zwięzłość jest w cenie, niemniej jednak trzeba się starać opisać o czym głównie jest praca, jak również co jest w tej pracy szczególnego, czego nie można znaleźć w innych. Zwykle abstrakt pisze się po napisaniu pracy, myśląc o takich kwestiach jak np. „jaki problem próbowano rozwiązać", „jaka była motywacja skupienia się nad tym problemem", „za pomocą jakich środków cel został osiągnięty". Wiele abstraktów różnego rodzaju prac można obejrzeć w Internecie.

Słowa kluczowe: praca dyplomowa, wzór, przewodnik

Abstract

Translation of your Polish abstract. Some leeway is allowed, but make sure it is a translation, not a completely different abstract. If you have a problem with English, ask your supervisor to help you translate. Machine translations (e.g. Google translate) are not good enough (yet...) to be acceptable.

Keywords: thesis, template, guide

[1] `http://www.editage.com/insights/how-to-write-an-effective-title-and-abstract-and-choose-a`

Wstęp

Krótkie omówienie tego, o czym będzie praca. (Czyli co zostanie w pracy powiedziane.)

Rzeczy które tu można ująć to np.

- mini-przewodnik po własnych wynikach, czy że zrobiono to, tamto i owamto
- motywacja do pracy, czyli dlaczego się tym zajęto i dlaczego masa rzeczy już na ten temat napisanych nie wystarczyła do szczęścia
- omówienie struktury pracy, czyli w tym rozdziale jest to, a tym owo
- historia badań na podobnymi tematami i aktualny stan wiedzy

Ilu autorów, tyle wstępów... Nie traktuj powyższych elementów jako obowiązkowe.

1

Rozdział o systemach przetwarzania obrazu nieopartych o głębokie uczenie maszynowe

(KOMENTARZ W tym rozdziale opisano systemy przetwarzania obrazu nieoparte o głębokie uczenie maszynowe – czyli w szczególności te na bazie algorytmu Viola-Jonesa oraz oparte o histogramy zorientowanych gradientów. Czy w tym rozdziale warto też od razu opisywać ataki? Na razie to robię, ale być może warto będzie zrobić jakieś przetasowanie. Czas pokaże. Czego NIE opisuję w tym rozdziale (póki co): dlaczego redukować w ogóle wymiarowość obrazów ("przekleństwo wymiarowości")). Co będę opisywać w tym rozdziale, ale będzie obejmować co najmniej te dwa algorytmy, więc nie opisuję dla każdego z osobna: jak wygląda pipeline'a przetwarzania danych (ekstrakcja cech-deskryptorów, trenowanie modelu, testowanie modelu)).

1.1 Histogramy zorientowanych gradientów

Histogramy zorientowanych gradientów (ang. *histograms of oriented gradients*, dalej: HOG) to deskryptory pozwalające na opisanie zawartości danego obrazu za pomocą wielkości i orientacji gradientów. Technika ta pozwala na redukcję wymiarowości obrazu oraz zniwelowanie wpływu lokalnych różnic na całość deskryptora[1].

1.1.1 Opis algorytmu

Algorytm tworzenia histogramów zorientowanych gradientów składa się z dwóch faz: wyliczenia gradientów oraz głosowania histogramów. Pierwsza pozwala na pozyskanie dla $n \times m$-wymiarowego obrazu w skali RGB dwóch $n \times m$-wymiarowych macierzy opisujących gradienty w tym obrazie. Druga korzysta z tych macierzy do wyznaczenia histogramów zorientowanych gradientów w celu dalszej redukcji wymiarowości obrazu. Fazy te zostały bardziej szczegółowo opisane poniżej.

Wyliczenie gradientów. Plikiem wejściowym jest analizowany obraz o wymiarach $n \times m$ pikseli, który jest przetwarzany do 8-bitowej skali szarości i dzielony na nakładające się komórki (ang. *cells*) ustalonej wielkości, wyznaczone wokół centralnego piksela danej komórki. Następnie dla każdej takiej komórki wylicza się wielkość ($M(x, y)$) i kąt ($\alpha(x, y)$) wektora gradientu:

$$M(x, y) = \sqrt{(x^2 + x^2)} \tag{1.1}$$

$$\alpha(x, y) = \arctan \frac{x}{y} \tag{1.2}$$

gdzie dla centralnego piksela danej komórki, znajdującego się w i-tej kolumnie j-tego wiersza, x i y to wartość bezwzględna różnicy wartości między najodleglejszymi pikselami odpowiednio w j-tym wierszu i i-tej kolumnie. Uzyskuje się w ten sposób dwie macierze $n \times m$ z wartościami odpowiadającymi wielkościom gradientów oraz ich kątom *modulo* 180°[2].

Głosowanie histogramu. Elementem wejściowym są macierze uzyskane w kroku pierwszym. Macierze są dzielone na nienakładające się bloki (ang. *blocks*). Następie w obrębie każdego bloku odbywa się głosowanie, którego wynikiem jest histogram danego bloku. Szczegółowy algorytm głosowania pokazano w pseudokodzie.

Efekt końcowy. Wynikiem działania zastosowanej metody są histogramy zorientowanych gradientów. Warto zauważyć, że użycie tej metody prowadzi do znacznego zmniejszenia wymiarowości danych. Przykładowo, można policzyć, że 24-bitowy obraz o wymiarach 64×128 pikseli, do którego opisania pierwotnie potrzeba by 24 576 wartości, może być podzielony na 27 bloków 16×16 pikseli, co daje 243 wartości w histogramach opisujące cały obraz. Jest to ponad stukrotne zmniejszenie liczby wartości opisujących obraz, które wciąż pozwala na stworzenie stosunkowo skutecznego deskryptora[1].

Algorithm 1 Głosowanie histogramu w bloku

Wejście: B_k – blok kątów w postaci listy i-elementowej, B_w – blok wielkości gradientów w postaci listy i-elementowej.

Wyjście: H – 9-elementowa lista definiująca histogram o klasach $0 - 20°$, $20 - 40°$, ..., $160 - 0°$.

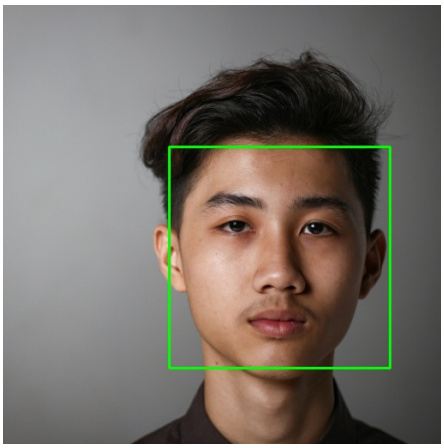
$H \leftarrow [0, 0, 0, 0, 0, 0, 0, 0, 0]$
for $i \leftarrow 1, n$ **do**
  $c \leftarrow \lfloor B_{k_i} \rfloor$
  $h \leftarrow c \div 20$
  if $c == B_{k_i}$ **then**
    $H[h] \leftarrow B_{w_i} \div 2$
    $H[h-1] \leftarrow B_{w_i} \div 2$
  else
    $H[h] \leftarrow B_{w_i}$

1.1.2 Klasyfikatory oparte o histogramy zorientowanych gradientów

Wyliczenie deskryptora dla wielu obrazów to pierwszy krok do stworzenia klasyfikatora danej cechy opartego o uczenie maszynowe. Popularność HOG przypada na wczesne lata 2000, kiedy znacznym poważaniem cieszyły się m.in. maszyny wektorów podpierających (ang. *Support Vector Machines*, dalej: SVM). Jest to model uczenia maszynowego, zaproponowany przez Vladimira Vapnika w 1995, pozwalający na wyznaczenie optymalnej hiperpłaszczyzny, która odziedzialałaby dane z różnych klas zachowując największy możliwy margines zaufania [3, 7]. Ten model został zastosowany również przez autorów metody histogramów zorientowanych gradientów, jednak jego bardziej szczegółowe omówienie wykracza poza zakres tej pracy i nie jest konieczne do zrozumienia sposobu działania metody ani możliwych dróg ataku na nią [1]. (KOMENTARZ Chyba że promotor bardzo będzie tego chciał albo zabraknie materiału.)

Wytrenowany klasyfikator może zostać następnie zastosowany do wykrywania danej cechy na różnych zdjęciach, także takich, które nie były użyte do jego wytworzenia. Przykładem klasyfikatora, którego można użyć do wykrycia twarzy *en face* jest ten załączony do biblioteki *dlib*, otwartoźródłowej biblioteki z narzędziami do uczenia maszynowego, napisanej w języku C++ i posiadającej też API do języka Python[4]. Poniżej zaprezentowano przykładową implementację wspomnianego klasyfikatora w celu rozpoznania twarzy na zdjęciu oraz efekt działania takiego skryptu.

Rysunek 1.1: Oryginalne zdjęcie.[6]



Rysunek 1.2: Efekt działania klasyfikatora.

Kod źródłowy 1.1: Wyszukiwanie twarzy z użyciem biblioteki dlib

```
import numpy as np
import cv2
import matplotlib.image as mpimg
import matplotlib.pyplot as plt
import dlib

image_colour = mpimg.imread('path_to_image.jpg') # load the image
image_gray = cv2.cvtColor(image_colour, cv2.COLOR_BGR2GRAY) # grayscale
    conversion
detector = dlib.get_frontal_face_detector() # load the classifier
faces = detector(image_gray, 0) # search for all potential faces in the
    grayscale image

res = image_colour.copy() # create a copy of the original image (to
    draw on it later)

for face in faces:
    # for each detected face, draw a green bounding box around it
    cv2.rectangle(res, (face.left(), face.top()), (face.right(), face.
        bottom()), (0, 255, 0), 2)

cv2.imwrite('output.jpg', cv2.cvtColor(res, cv2.COLOR_BGR2RGB)) # save
    results
```

Rysunek 1.3: Atak przez unikanie. Źródło zdjęcia: opracowanie własne.

1.1.3 Metody ataku

Ze względu na ich stosunkowo niski poziom skomplikowania i wysoką intepretowalność w porównaniu do modeli opartych o głębokie uczenie maszynowe istnieją metody pozwalające na wizualizację deskryptorów HOG. Metody te, poza ułatwianiem pracy nad debugowaniem istniejących modeli, mogą także służyć do opracowywania metod ataku. Przegląd istniejącej literatury wskazuje, że rozwiązania oparte o HOG są podatne na dwa typy ataków: ataki przez unikanie (ang. *evasion attacks*) oraz ataki przez zatrucie (ang. *poisoning attacks*)[8, 5]. (KOMENTARZ: szersza charakterystyka typów ataków oparta o proponowaną przez NIST nomenkulaturę znajdzie się gdzieś we wstępnym rozdziale.)

Ataki przez unikanie. W przypadku HOG ataki przez unikanie polegają na przesłonięciu w obiekcie fragmentu lub fragmentów odpowiadających blokom o znaczących wartościach deskryptora. Przykładowo, w przypadku opartego o HOG detektora twarzy we wspomnianej bibliotece dlib, skuteczność wykrywania znacząco spada dla twarzy w ciemnych okularach: dla zbioru, na którym trenowany był dlibowski *frontal face detector* wynosi 91% dla ogółu zdjęć oraz 57% dla zdjęć osób w ciemnych okularach zasłaniających oczy (niezależnie od kształtu okularów).[1] Podobny efekt można uzyskać dzięki założeniu np. maseczki albo komina zasłaniającego dolną część twarzy[4, 5].

Ataki przez zatrucie. Innym sposobem na przechytrzenie systemu przetwarzania obrazu jest zwizualizowanie wyuczonego detektora (jeśli mamy do niego

[1].(KOMENTARZ: osoba pisząca wyjęła sobie te dane z d...obrze napisanego skryptu, który przeiterował przez wszystkie obrazy oraz przez podgrupę obrazów z osobami w dość ciemnych okularach. Czy to materiał do jakiegoś appendixu?)

dostęp) albo histogramu gradientów dla pozytywnego przykładu, a następnie użycie go do „zatrucia” materiału testowego poprzez np. zapozowanie w ubraniu z nadrukiem pokazującym tę wizualizację. W takim przypadku system popełniałby błędy pierwszego rodzaju, odnajdując poszukiwane obiekty na zdjęciach, które mogłyby ich wcale nie zawierać. Ataki przez zatrucie mogą też być połączone z atakami przez unikanie: przykładowo, w przypadku systemów wymagających, by na zdjęciu znalazła się jakaś twarz i docinających do niej zdjęcie, założenie maseczki z „zatrutym” wzorem skutkowałoby jednoczesnym uniknięciem wykrycia właściwej twarzy na zdjęciu i zapamiętaniem materiału błędnego z punktu widzenia człowieka[5]. (KOMENTARZ: tutaj też będzie twarzowe zdjęcie, ale ze względu na braki drukarkowe pojawi się później.)

1.2 Algorytm Violi-Jonesa

Tutaj będzie opisanych na kolejnych kilka stron algorytm Violi i Jonesa.

Bibliografia

[1] N. Dalal and B. Triggs. *Histograms of oriented gradients for human detection*, volume 1, pages 886–893 vol. 1. 2005.

[2] R. Gonzalez and year=2008 edition=3rd isbn=9780131687288 publisher=Pearson R. Woods, booktitle=Digital Image Prcessing.

[3] N. Jankowski. *Ontogeniczne sieci neuronowe. O sieciach zmieniających swoją strukturę*. 1st edition, 2003.

[4] Davis E. King. dlib. a toolkit for making real world machine learning and data analysis applications in c++.

[5] Bruce MacDonald. Fooling facial detection with fashion.

[6] Imansyah Muhamad Putera. boy's face. `https://unsplash.com/photos/n4KewLKFOZw`, 2020. Dostęp: 2020-06-17.

[7] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin, Heidelberg, 1995.

[8] C. Vondrick, A. Khosla, T. Malisiewicz, and A. Torralba. Hoggles: Visualizing object detection features. In *2013 IEEE International Conference on Computer Vision*, 2013.

Dodatek A: Cośtam dodatkowego

Dodatki zawierają treści, których zrozumienie nie jest konieczne do zrozumienia pracy i które mogłyby niepotrzebnie zajmować miejsce. Czasem są to listingi programów – może np. w pracy wystarczą krótkie fragmenty do których się akurat odnosimy, a w dodatku może być cały kod źródłowy, itp.

Akronimy

KISS Keep It Simple Stupid

IT Information Technology