Projekt indywidualny - SVD

Piotr Klepacki

March 8, 2023

1 Wstęp teoretyczny do SVD

SVD czyli Rozkład Według Wartości Osobliwych (ang. Singular Value Decomposition) to metoda faktoryzacji macierzy. Jej wynikiem są trzy macierze.

$$X = U\Sigma V^T$$

U oraz V są macierzami ortagonalnymi, czyli spełniającymi następujący warunek:

$$AA^T = A^T A = I$$

$$A \in M_n R$$

gdzie I jest macierzą jednostkową. U jest macierzą złożoną z kolumn, które są wektorami własnymi o rozmiarach takich samych jak kolumny macierzy początkowej. Σ jest macierzą diagonalną. Jej elementy są posegregowane hierarchicznie. V^T jest macierzą, której wiersze zawierają informację, o tym w jakich proprocjach należy połączyć wektor własny z kolumny U, aby otrzymać odpowiadającą kolumne zawartą w pierwotnej macierzy.
Pomimo że macierz Σ może mieć większy rozmiar niż X, posiada ona tylko tyle elementów niezerowych, co ilość kolumn X. Oznacza to, że wynik mnożenia kolumn U o indeksach większych niż m, zawsze będzie równy zero. Z tego powodu przeważnie używa się zredukowanej wersji SVD zwanego Economy SVD, w której używa się wyłącznie części macierzy, które po przemnożeniu bedą niezerowe.

$$\text{Economy SVD} = \hat{U}\hat{\Sigma}V^T$$

Ze względu na hierarchiczne ułożenie macierzy $\hat{U}\hat{\Sigma}V^T$, możliwe jest uzyskanie wyników z duża dokładnością, bez przeprowadzania operacji na wszystkich wartościach szczególnych podczas rekonstrukcji macierzy bazowej.
Wybieramy taki rząd r, że r < m. Mnożymy tylko r pierwszych kolumn U, przez macierz Σ o rozmiarach r na r oraz przez r pierwszych wierszy V^T.

$$\text{Aproksymacja r} = \widetilde{U}\widetilde{\Sigma}\widetilde{V^T}$$

Odpowiednie dobranie wartości r, pozwala na znaczne skrócenie czasu obliczeń i rozmiarów produktów SVD, z nieznacznym spadkiem dokładności wyniku. Oczywiście dobranie prawidłowego r nie jest proste i różni się w zależności od tego jakie informacje zawiera macierz. Różnić się może zarówno procentowa wartość informacji o macierzy dla

tego samego r jak i względna wartość wynikająca z okoliczności zastosowania algorytmu. Dla przykładu, dla tego samego zdjęcia możemy przyjąć różne r w zależności od tego, czy zależy nam bardziej na zaoszczędzonym miejscu czy zachowaniu szczegółów obrazu.

2 Przegląd i opis możliwości wykorzystania algorytmu SVD w przetwarzaniu obrazów.

SVD znajduje zastosowanie w wielu algorytach przetwarzania obrazów. Przez przedstawienie obrazu w postaci macierzy, a następnie przekształcenia jej w wektor, który umieszczany jest w macierzy złożonej z wektorów innych zdjęć, operacje na nich i późniejsza rekonstrukcja są bardzo proste. Pozwala to na zastosowanie dosyć prostych programów do realizacji przykładów opisanych poniżej, co jednak wiąże się z dość dużym obciążeniem maszyny, przez ilość obliczeń wymaganych do przeprowadzenia na większych zbiorach danych.

2.1 Redukcja rozmiaru obrazu

Tak jak wspomniałem na początku tej sekcji, wszystkie obrazy mogą być przedstawione w formie macierzy. Po rozkładzie SVD otrzymujemy wartości szczególne, które ułożone są hierarchicznie, tj. każda kolejna wartość jest mniej istotna dla reperezentacji macierzy początkowej. Możliwa jest więc redukcja rozmiarów obrazu, przez wybranie niższego rzędu r. Jako że w pierwszej kolejności odrzucamy wartości szczególne mające mniejsze znaczenie, możliwe jest znaczne zmniejszenie rozmiaru obrazu przy stosunkowo dokładnym go zachowaniu. Nie jest to jednak metoda całkowicie uniwersalna i zastosowanie stałej wartości r nie będzie zawsze przynosiło satysfakcjonujących efektów. Przykłady takiej sytuacji przedstawione są w sekcji 4.1.

2.2 Algorytmy rozpoznawania twarzy

Jednym z najbardziej praktycznych zastosowań SVD jest jego użycie w PCA - Analizie głównych składowych (ang. principal component analysis). Wynikiem analizy jest hierarchiczny system współrzędnych reprezentujący wielowymiarowy zbiór danych. Przed zastosowaniem SVD, od macierzy X odejmuje się wartości średnie i wyznacza macierz kowariancji. Wynikowe główne składowe są ze sobą nieskorelowane ale posiadają maksymalną korelację z danymi. Podobnie jak w SVD danymi wejściowymi jest macierz o wymiarach *nxm*, gdzie m jest ilością rozpatrywanych pomiarów lub, jak w naszym przypadku zdjęć, a n ilością własności opisujących dane zdjęcie, a więc odwrotnie jak w SVD. Oznacza to, że każde zdjęcie przekształcone jest z macierzy w wektor kolumnowy. Schemat PCA:

Obliczenie średniego wiersza X:

$$\tilde{x_j} = \frac{1}{n}\sum_{i=1}^{n} Xij$$

Macierz średnich

$$\widetilde{X} = \begin{bmatrix} 1 \\ \cdot \\ \cdot \\ 1 \end{bmatrix} \widetilde{x}$$

Wyznaczanie macierzy odchyleń B

$$B = X - \widetilde{X}$$

Macierz kowariancji C

$$C = \frac{1}{n-1} BB$$

Dekompozycja C

$$CV = VD$$

gdzie V jest macierzą wektorów własnych, a D macierzą wartości szczególnych.
Macierz głównych składowych T jest zdefiniowana jako

$$T = BV$$

Możliwe jest obliczenie T przy pomocy obliczenia rozkładu SVD z B

$$B = U\Sigma V^T$$

$$\mathrm{T} = \mathrm{U}\ \Sigma$$

2.3 Usuwanie przysłonięć ze zdjęć

Do usuwania przysłonięć ze zdjęć posłużyłem się RPCA, będącym modyfikacją PCA. PCA jako model posługujący się metodą namniejszych kwadratów jest podatny na błędy i duże odchylenia od typowych danych. RPCA został opracowany w celu rozwiązania tego problemu, przez dekompozycje macierzy X na macierz o niskim rzędzie L oraz macierz rzadką S, który zawiera wartości nietypowe. Poniżej przedstawiłem przykład implementacji dekompozycji, zaprezentowany w książce "Machine Learning, Dynamical Systems and Control" autorstwa Stevena L. Bruntona oraz J. Nathana Kutza.

```
def shrink(X, tau):
Y = np.abs(X) - tau
return np.sign(X) * np.maximum(Y, np.zeros_like(Y))

def SVT(X, tau):
U, S, VT = np.linalg.svd(X, full_matrices=0)
out = U @ np.diag(shrink(S, tau)) @ VT
return out

def RPCA(X):
```

```
n1, n2 = X.shape
mu = n1 * n2 / (4 * np.sum(np.abs(X.reshape(-1))))
lambd = 1 / np.sqrt(np.maximum(n1, n2))
thresh = 10 ** (-7) * np.linalg.norm(X)
while (np.linalg.norm(X - L - S) > thresh) and (count < 1000):
L = SVT(X - S + (1 / mu) * Y, 1 / mu)
S = shrink(X - L + (1 / mu) * Y, lambd / mu)
Y = Y + mu * (X - L - S)
return L, S
```

3 Własna implementacja algorytmu SVD.

W tej sekcji omówię mój proces implementacji algorytmu SVD przez dokładne wykonanie operacji matematycznych.

3.1 Opis wybranego algorytmu

Do własnej implementacji SVD wybrałem metodę Iteracji Ortogonalnej. W założeniu algorytm jest prosty. Jest jednak również naiwny, i przy większych rozmiarach macierzy wymaga dużej ilości obliczeń. Sprawia to, że nie jest on optymalny do zastosowania w przetwarzaniu obrazów, które potrafią mieć wymiaru rzędu kilku tysięcy pikseli. Sprawdza się jednak dla mniejszych macierzy i pojednyczych obliczeń, a zarazem jest prosty, co sprawia że był bardzo przydatny w zrozumieniu działania i zastosowań SVD. Metoda ta bazuje na założeniu, że wszystkie pozostałe wektory własne są ortogonalne do dominującego wektora własnego. Jest wersją blokową Power Method. Algorytm przedstawia się następująco:

1. wybierz Q_0 takie że $Q_0 Q_0^T = I$
2. dla k=1,2,..:
   $Z_k = AQ_{k-1}$
   $Q_k R_k = Z_k$ - dekompozycja QR
3. gdzie R_k: $R_k = Q_k^T Z_k = Q_k^T AQ_{k-1}$

4 Testy wykorzystania algorytmu SVD do min. 3 problemów z dziedziny przetwarzania obrazów.

4.1 Redukcja rozmiaru obrazu

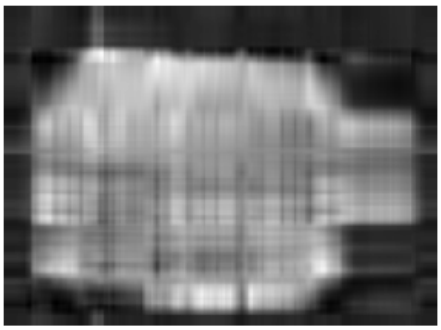
Testy algorytmu do redukcji rozmiaru obrazu przeprowadziłem na 3 zdjęciach o różnych rozdzielczościach i przedstawiające różne obiekty. Głownym celem takiego doboru zdjęć testowych, jest porównanie skuteczności kompresji w zależności od kompozycji i rozmiarów obrazu.

4.1.1 Obraz o małej rozdzielczości

Obiekt przedstawia rysunek małego domku o rozdzielczości 200 x 146 pi.

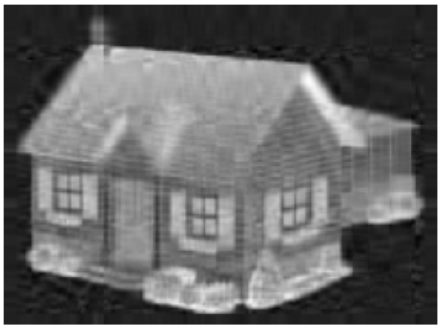
Rekonstrukcja obrazu dla różnych ilości użytych wartości szczególnych r.

r = 5

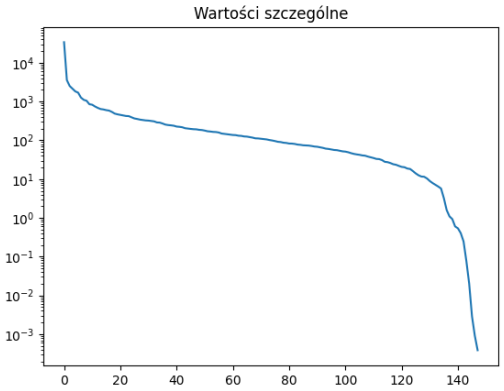


r = 100

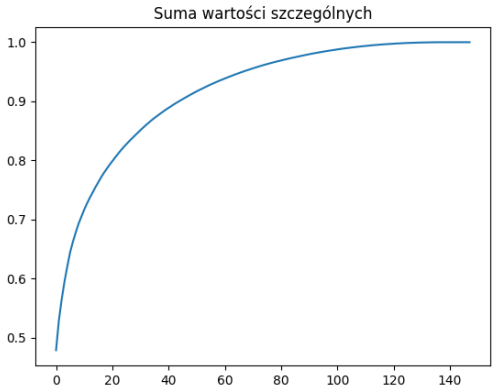
r = 20



r = 200



(a) Wykres przedstawiający wartość szczególną w zależności od rzędu r

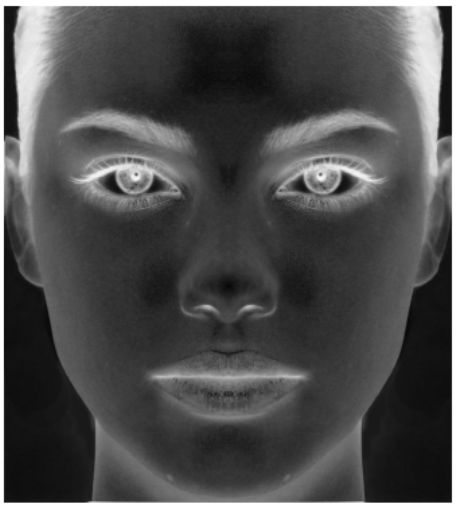


(b) Wykres przedstawiający sume wartości szczególnych mniejszych od rzędu r

Zarówno rekonstrukcja dla r równego 100, jak i 200 przyniosła bardzo zbliżone efekty. Patrząc na wykresy (a) i (b) nie jest to zaskakujący wynik, jako że dla r = 100 suma wartości szczególnych jest bliska 1. Warto zaznaczyć, że w rekonstrukcji dla r = 200 jedynie pierwsze 146 wartości jest niezerowych, co oznacza że obrazek jest identyczny do jego wejściowej wersji. Dla r = 20 rysunek jest rozmazany i niezbyt czytelny, ale jednoczeście wystarczająco wyraźny żeby rozpoznać co przedstawia, a nawet wyróżnić szeczgóły. Dla r = 5 widoczna jest jedynie plama jaśniejszych pikseli w kształcie domku, chociaż znając pierwotny obrazek możemy zauważyć kontury niektórych szczegółów, np roślinności przed domkiem.

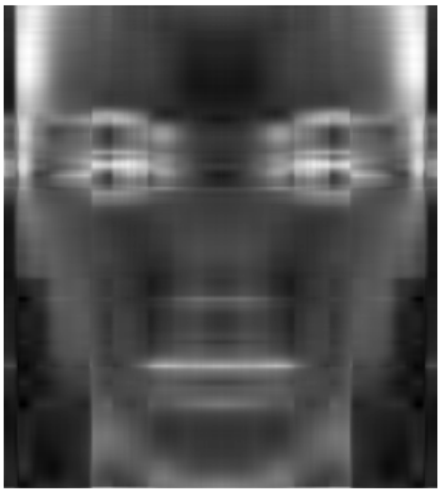
4.1.2 Obraz o średniej rozdzielczości

Obiekt przedstawia twarz o rozdzielczości 864 x 960 pi.



Rekonstrukcja obrazu dla różnych ilości użytych wartości szczególnych r.

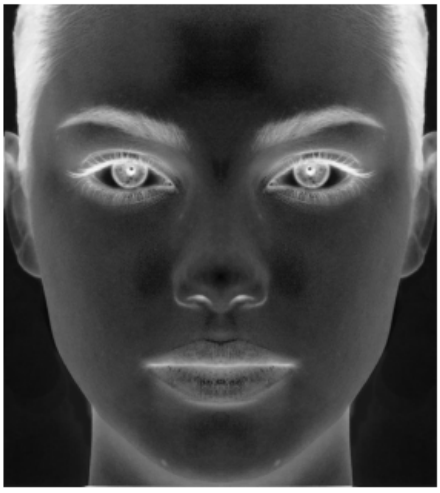
r = 5

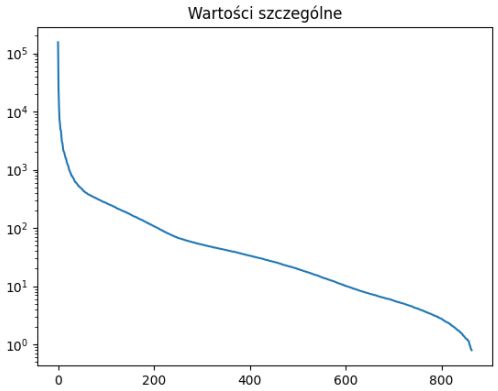


r = 100

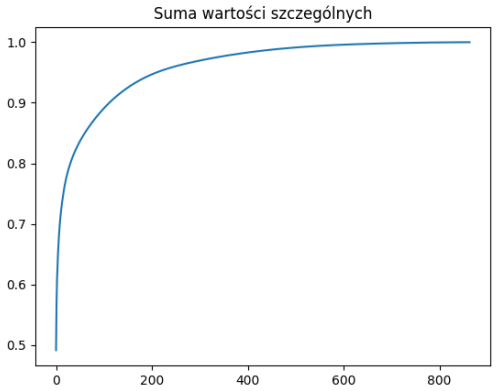
r = 20

r = 200





(a) Wykres przedstawiający wartość szczególną w zależności od rzędu r

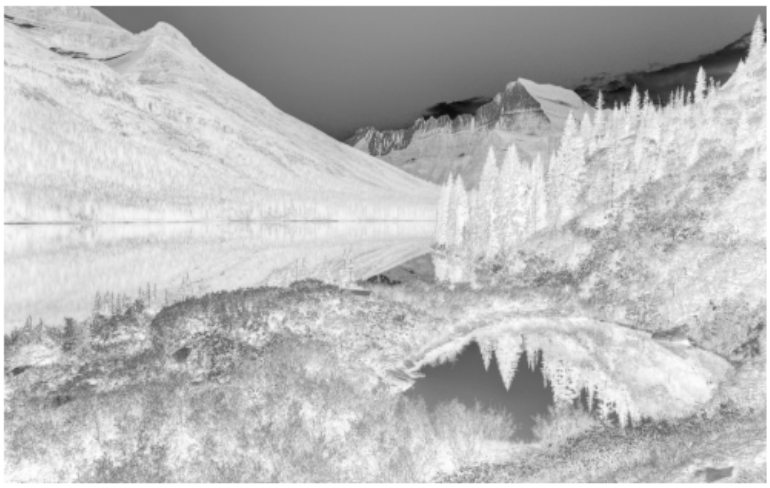


(b) Wykres przedstawiający sume wartości szczególnych mniejszych od rzędu r

Dla zdjęcia twarzy suma wartości szczególnych (b) bardzo szybko rośnie dla początkowych 50 - 100 rzędów r, po czym tempo przyrostu stopniowo maleje. Odzwierciedlone to jest w rekonstrukcji obrazu, gdzie różnice pomiędzy zdjęciami dla r = 100 oraz r = 200, są niezauważalne. Dla r = 20 twarz jest jedynie nieco rozmazana, ale wszystkie szczegóły są wyraźne, zwłaszcza te o pikselach w jaśniejszych odcieniach jak oczy czy brwi. Dla r = 5 obraz jest oczywiście mało szczegółowy i złożony z dużych obszarów jednakowego koloru pikseli, jednak można wyrożnić zarówno zarys twarzy, jak i jej poszczególne elementy.

4.1.3 Obraz o dużej rozdzielczości

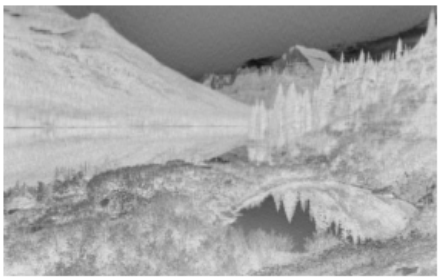
Obiekt przedstawia rysunek krajobrazu o rozdzielczości 2560 x 1600 pi.



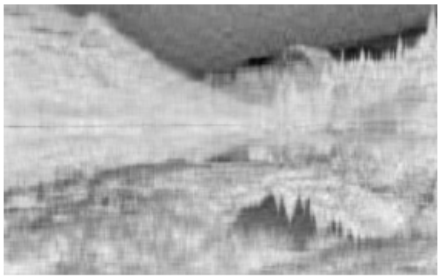
Rekonstrukcja obrazu dla różnych ilości użytych wartości szczególnych r.

r = 5

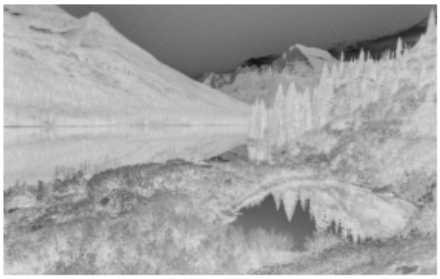
r = 100

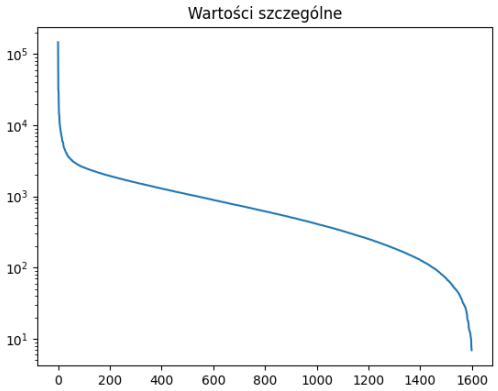


r = 20

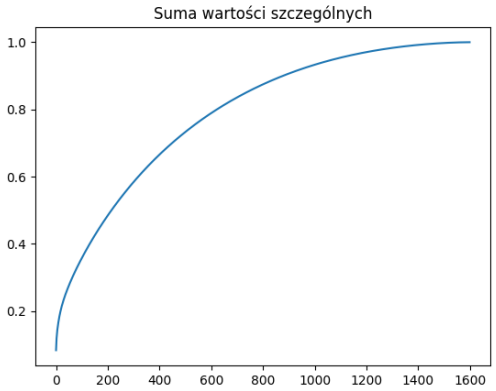


r = 200





(a) Wykres przedstawiający wartość szczególną w zależności od rzędu r

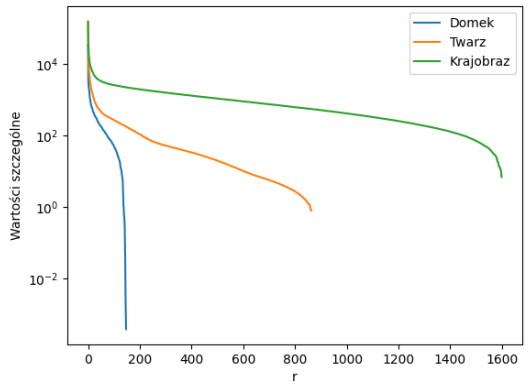


(b) Wykres przedstawiający sume wartości szczególnych mniejszych od rzędu r

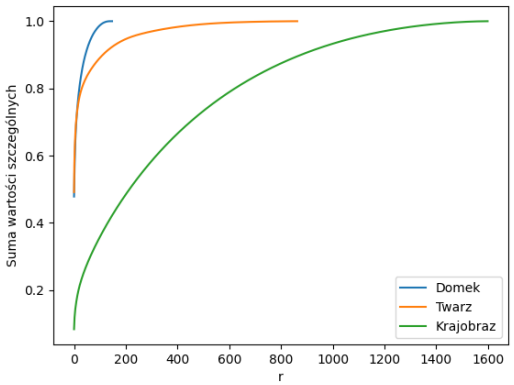
Na wykresie (a) dla ostatniego rozpatrywanego obrazu, widzimy że spadek wartości szczególnych jest największy dla pierwszych i ostatnich 50 rzędów r. Wykres sumy wartości szczególnych (b) jest bardziej spłaszczony, a tempo przyrostu powolniejsze, co sprawia że wybór prawidłowego rzędu r jest problematyczny dla tego przypadku. Jedynie dla tego obrazu różnice w rekonstrukcji pomiędzy r = 100, a r = 200 są widoczne. Oba obrazy wynikowe dobrze odzwierciedlają zdjęcie początkowe, różni je jedynie ostrość szczegółow. Jest to nabardziej widoczne w jakości zarysu chatki na tle nieba. Dla r = 20 możliwe jest rozpoznanie przedstawionego krajobrazu i jego poszczególnych elementów, ale dla r = 5 właściwie niemożliwe jest stwierdzenie co przedstawiał obraz początkowy.

4.1.4 Wnioski

Jak zaznaczyłem we wstępie do tej sekcji, zdjęcia różniły się zarówno rozdzielczością, jak i obiektem który przedstawiały. Chociaż algorytm nie odróżnia czy zdjęcie przedstawia twarz, czy samochód to sama kompozycja zdjęcia jest ważna.



(a) Wykres przedstawiający wartość szczególną w zależności od rzędu r



(b) Wykres przedstawiający sume wartości szczególnych mniejszych od rzędu r

Jak widać na powyższych wykresach, funkcje zarówno wartości szczególnych, jak i ich sumy, przebiegają podobnie. Największą różnicą jest szybkość spadku znaczenia danego rzędu r dla rekonstrukcji obrazu bazowego. Jest to prawdopodobnie spowodowane przez:

1. Rozmiar obrazu
   Najbardziej oczywistym jest wpływ rozdzielczości, a więc i rozmiaru macierzy przechowywującej informacje o obrazie na wymagany rząd rekonstrukcji. Z tego też powodu różne są zarówno długości wykresów, jak i wartości funkcji dla r = 1, przez różnice w stosunku danego r do całkowitego rozmiaru macierzy.

2. Ilość szczegółow
   Wpływ ilości szczegółów jest najbardziej zauważalny porównując zdjecia krajobrazu z pozostałymi obrazami. Posiada on znacznie więcej pojedynczych elementów wyróżniających się na tle nieba, pojedyncze drzewa, zarysy gór, ukształtowanie podłoża. Na dwóch pierwszych zdjęciach, ilość szczegółów jest znacznie mniejsza.

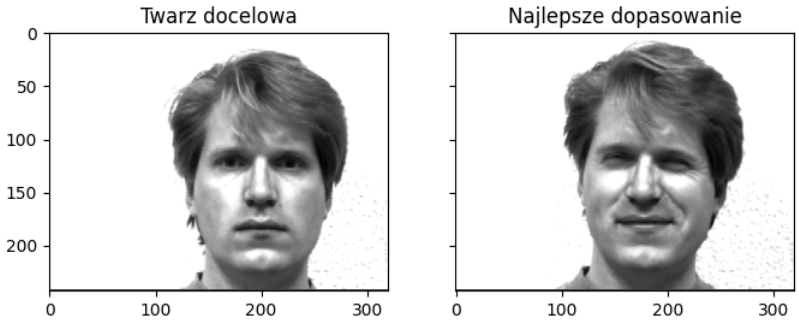
3. Rozmiar tła i kolor obiektu
   Ostatni powód jest bezpośrednio związany z poprzednim podpunktem. Im więcej pikseli sąsiaduje z pikselami o równej lub zbliżonej wartości, tym mniejszy rząd r jest potrzebny do ich prawidłowej reprezentacji. To jest główny powód tak szybkiego wzrostu sumy wartości szczególnych dla zdjęcia twarzy. Na obrazie znajduje się sporo jednolitych pikseli tła, a sama twarz jest również ciemna. Jedynymi jasnymi elementami są szczegóły twarzy, ale przez tak wyraźną różnicę w kolorze, są one widoczne nawet dla bardzo niskiego rzędu r. Dlatego też wartość funkcji dla krajobrazu rośnie tak powoli. Zawiera on zbyt wiele szczegółów w zupełnie innych od sąsiednich pikseli barwach.

Powyższe przykłady dobrze obrazują poruszony wcześniej problem. Wybór prawidłowego rzędu r może stanowić znaczącą przeszkodę, szczególnie jeżeli operacje przeprowadzane są na zróżnicowanym zbiorze danych. Pomimo tego, możliwy jest wybór uniwersalnej wartości r, jeżeli zajdzie taka potrzeba. Dla przykładu, wszystkie 3 zdjęcia są rekonstruowane w zadowalającej jakości biorąc pod uwagę tylko pierwsze 100 rzędów r. Jest to szczególnie dobry wynik dla większych obrazów, których procentowa redukcja rozmiaru jest największa.

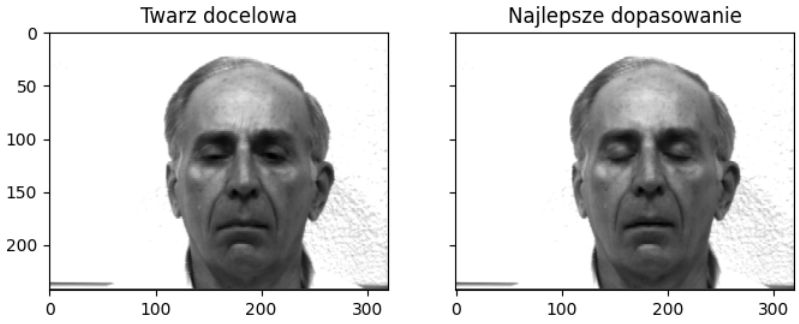
4.2 Algorytm rozpoznawania twarzy

Do przeprowadzenia testów i badań na moim algorytmie użyłem bazy twarzy YaleFacesB stworzoną przez uniwersytet Yale. Zawiera on po 11 zdjęć 15 osób, łącznie 165 portretów. Algorytm przyjmuje zdjęcie, które chcemy dopasować do osoby i zwraca je wraz ze zdjęciem najbadziej zbliżonym do zadnego wraz z informacją czy oba należą do tej samej osoby. Poniżej przedstawione są przykłady działania programu z różnymi efektami.

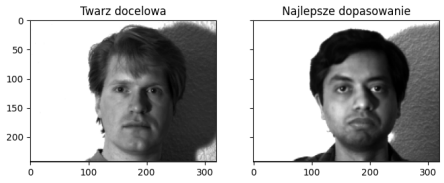
4.2.1 Efekty działania programu



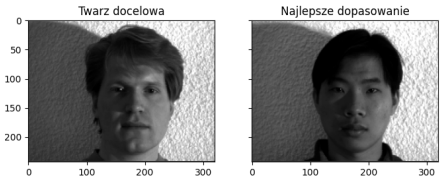
Sukces dopasowania dla zdjęć jednej osoby



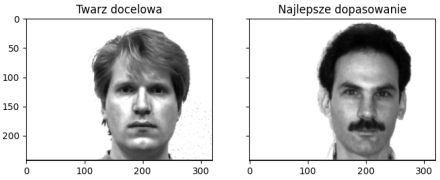
Sukces dopasowania dla wszystkich zdjęć



(a) Pierwsze błędnie dopasowana osoba



(b) Druga błędnie dobrana osoba

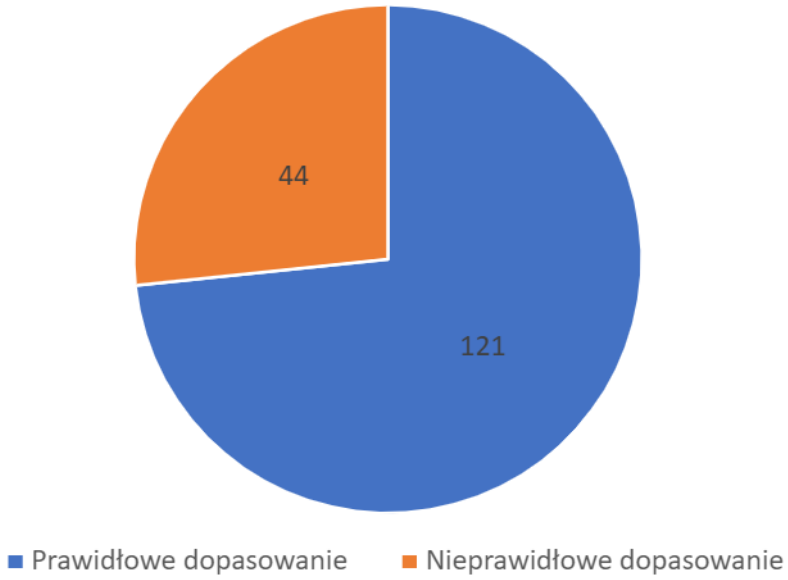


(c) Trzecia błędnie dobrana osoba

Błędne dopasowanie dla wszystkich zdjęć

4.2.2 Wnioski

Jak widać na zamieszonym przykładzie, złe dopasowanie następuje w przeważającej ilości przypadków dla zdjęć zrobionych w tych samych warunkach tj. obecność okularów, ułożenie źródła światła oraz cienia, zwrócenie głowy w tą samą stronę lub ten sam wyraz twarzy. Jest to efektem tego, że w użytej przezemnie bazie zdjęć, każde z 11 portretów miało te same, wyżej wspomniane i identyczne dla wszystkich osób założenia. Drugi powód jest bezpośrednio powiązany z pierwszym. Program tak naprawdę nie rozpoznaje twarzy. Szuka on najbardziej podobnego zdjęcia wśród tych rozpatrywanych podczas użycia PCA. Oznacza to, ze tło może mieć większy wpływ na dobranie zdjęć niż same cechy twarzy. Możliwym jest oczywiście zastosowanie innych metod do wycinania tła, wtedy jednak spotykamy się z problemem przedstawiania nieregularnego kształtu twarzy w postaci macierzy. Prawdopodobnie najlepszym i najprostrzym sposobem na usprawnienie skuteczności, byłoby zmniejszenie rozmiarów zdjęcia, aby każda twarz nadal się w nim mieściła, jednocześnie usuwając fragmenty tła.
Skuteczność algorytmu została sprawdzona przez wywołanie programu na każdym zdjęciu z używanej bazy. Po podsumowaniu efektów działania programu wyniki przedstawiają się następująco:



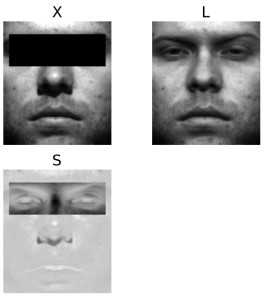
Wykres przedstawiający skuteczność działania programu

Zgodnie z powyższym wykresem skuteczność programu wynosi 73,33%. Z jednej strony nie jest to wynik bliski 100%, ale należy wziąć pod uwagę okoliczności, o których pisałem wcześniej. Wynik ten pokazuje, że w połączeniu z obróbką i selekcją zdjęć, powyżej omówiony algorytm może być bardzo przydatny w systemach rozpoznawania twarzy, operując jedynie na ściśle określonych i w gruncie rzeczy schematycznie prostych operacjach matematycznych. Największym wymaganiem jest jednak potrzebna moc obliczeniowa. Operowanie na takiej ilości zdjęć często sprawiało problemy mojemu niezbyt imponującemu laptopowi. Wielokrotnie spotkałem się z sytuacją, w której musiałem ponownie przeprowadzić jakąś partie testów, po błędzie w programie spowodowanym zbyt małą ilością pamięci RAM. Jest to jednak prawdopodobnie pomijalny problem w przypadku pracy na wydajniejszych maszynach lub w chmurze.

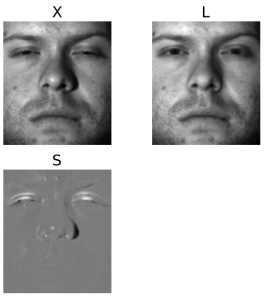
4.3 Usuwanie przysłonięć z obrazów

Do testów w tej sekcji użyłem innego zbioru zdjęć uniwerystetu Yale, Yale extended faces b. W przeciwieństwie do zdjęć z sekcji 4.2, wszystkie portrety zostały wykonane z tej samej pozycji, a jedyną różnicą pomiędzy nimi jest położenie źródła światła. Postanowiłem również ograniczyć się do 12 zdjęć jednej z osób, z powodu bardzo długiego, czasami kilkuminutowego, czekania na wynik. Zdjęcia podzieliłem na 3 grupy w zależności od wielkości i lokacji przysłonięcia. W każdej z grup, przysłonięcie w postaci czarnego prostokątu występuje tylko na jednym i tym samym zdjęciu.

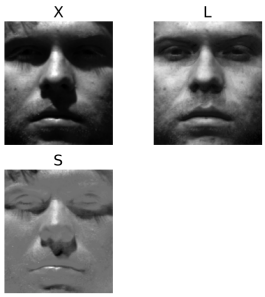
4.3.1 Efekty działania programu



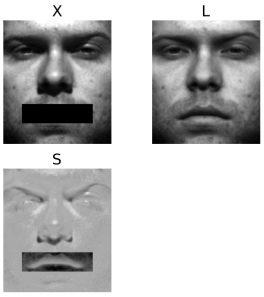
(a) Zdjęcie z przysłonięciem

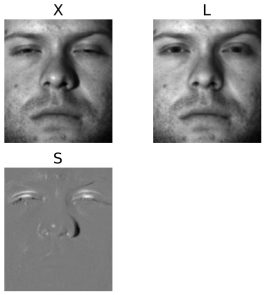


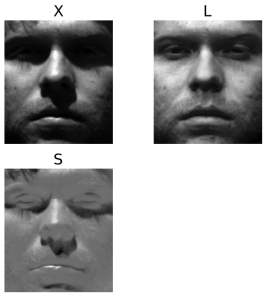
(b) Przykładowe zdjęcie bez przysłonięcia (nr. 15)

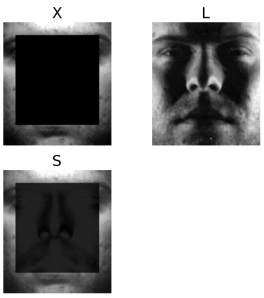


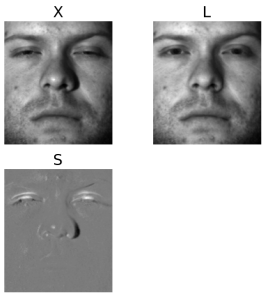
(c) Przykładowe zdjęcie bez przysłonięcia (nr. 18)

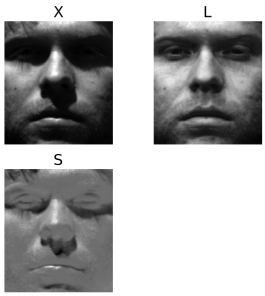






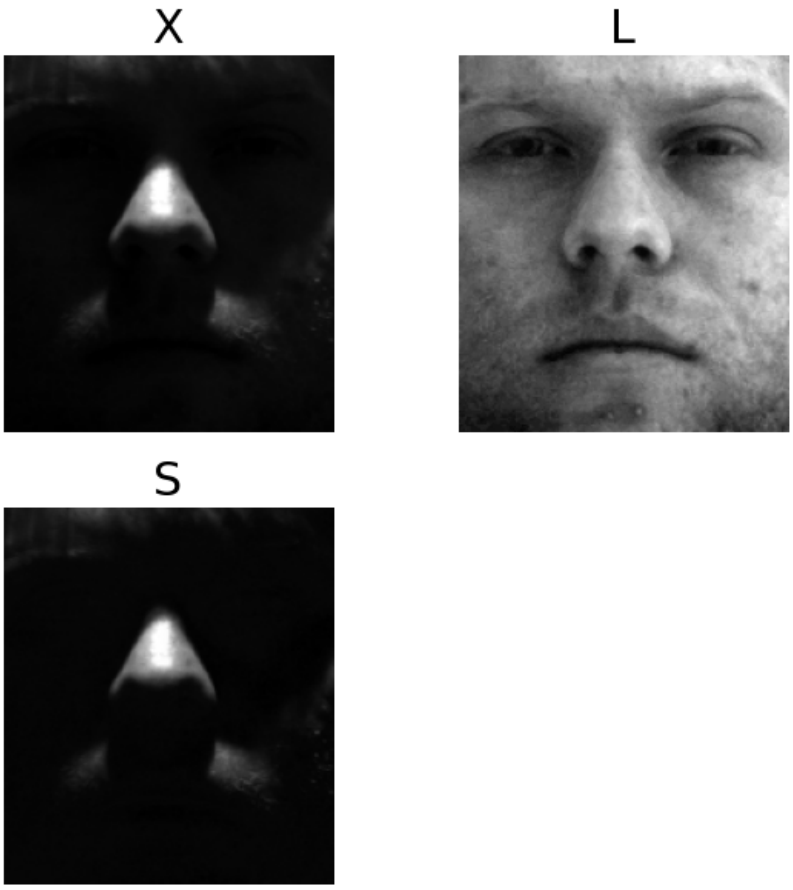






Nazwy zdjęć wynikowych są zgodne z oznaczeniami macierzy zamieszczonymi w sekcji 2.3. Program poprawnie usunął przysłonięcia we wszystkich trzech przypadkach. Oczywiście rekonstrukcje zdjęć przysłoniętych różnią się od siebie, jednak bez trudu można rozpoznać, że jest to ta sama twarz. Najciekawszy jest przypadek, w którym czarny prostokąt znajduje się na praktycznie całej twarzy. Obraz stworzony z wartości w macierzy niskiego rzędu L jest wyraźnie ciemniejszy. Z kolei w zdjęciu z macierzy rzadkiej S, wyraźnie widoczne są elementy twarzy, które nie były przykryte. Można zauważyć również wpływ przysłonięcia na resztę zdjęć. Przykładem na to jest lekko widoczny prostokąt w zdjęciu nr.18, w rzędzie z przysłoniętymi oczami. Kolejnym ciekawym wynikiem który napotkałem, było działanie algorytmu na zdjęciu, które juz pierwotnie było ciemne przez

małą ilość swiatła.



Wynik działaniu programu na zdjęciu z małą ilością swiatła (nr.4)

Chociaż dla ludzkiego oka rozpoznanie zarysu twarzy w tych warunkach jest problematyczne, program jest w stanie bez trudu ją odtworzyć i to z lepszymi efektami niż w przypadku mojego sztucznego jej przysłaniania.

4.3.2 Wnioski

Przy pomocy algorytmu RPCA udało się celnie usunąć niechciane obiekty z każdego zdjecia. Dzięki wykadrowaniu zdjęć, efekty działania programu są dobrze widoczne, nie tylko na podstawie zdjęcia L, ale również S, złożonego z wartości nietypowych. Zdjęcia S pokazują, które elementy twarzy sa najbardziej zmienne, są to usta, nos i oczy. Dzięki wykadrowaniu twarzy, jej zarys nie jest praktycznie nigdy widoczny. Jednak według mnie algorytm RPCA miał najciekawszy efekt nie na zdjęciach sztucznie przezemnie przysłoniętych, a na zdjęciu nr.4. Podczas gdy zakrycie twarzy czarnym prostokątem,

poskutkowało pociemnieniem zdjęcia wynikowego, drobne różnice w kolorze pikseli zarysowujące twarz pomimo nikłej obecności światła, wystarczyły do rekonstrukcji zdjęcia praktycznie identycznego do tego zrobionego w dobrym świetle. Ukazuje to przydatność algorytmu choćby w próbach odtworzenia obiektów mniej widocznych na monitoringu przy utrudnionej widoczności lub częściowej korupcji obrazu.

5 Wnioski końcowe

Praca nad projektem pozwoliła mi na pogłebienie wiedzy nie tylko informatycznej, ale również matematycznej. Sam fakt, że zarówno SVD jak i PCA zostały opracowane wiele lat przed zaimplementowaniem ich w dziedzinach komputerowych jest niesamowicie imponujący. Podczas realizacji projektu najwięcej czasu spędziłem ucząc się i badając podane zagadnienia, co było ciekawą odmianą od projektów realizowanych we wcześniejszym toku studiów. Pomimo tego, że w moim projekcie badałem jedynie trzy zastosowania SVD, byłem pod dużym wrażeniem jego przydatności oraz skuteczności. Wszystkie testy przeprowadzałem na zdjęciach, ale wyniki działania na macierzach zawierających inne informacje, byłyby z pewnością równie, jeżeli nie bardziej, udane. Nie dziwi mnie więc, że algorytm znajduje zastosowanie w wielu dziedzinach.

6 Bibliografia

https://towardsdatascience.com/simple-svd-algorithms-13291ad2eef2
http://databookuw.com/databook.pdf
http://sepwww.stanford.edu/data/media/public/docs/sep73/ray1/paper_html
http://databookuw.com/
https://machinelearningmastery.com/face-recognition-using-principal-component-analysis/