WOJSKOWA AKADEMIA TECHNICZNA

WYDZIAŁ CYBERNETYKI

Biometryczne systemy rozpoznawania

Sposoby identyfikacji użytkowników na podstawie tęczówki

Prowadzący: dr inż. Leszek Grad

Autor: sierż. pchor. Damian Krata

Grupa: I7D1S4

Data: 24.05.2018

1. Zadanie

Celem zadania laboratoryjnego było zapoznanie się ze sposobami identyfikacji użytkowników na podstawie tęczówki. W ramach tego zagadnienia należało zapoznać się z mechanizmami wykorzystującymi transformatę kosinusową (DCT) oraz z sieciami neuronowymi. Za pomocą skryptów napisanych w programie MATLAB zaimplementowano oraz przetestowano mechanizm rozpoznawania wzorców dla bazy tęczówek dostarczonej przez prowadzącego.

2. Wstęp teoretyczny

Tęczówka (łac. iris) – element budowy oka, nieprzezroczysta tarczka stanowiąca

przednią część błony naczyniówkowej. W centrum zawiera otwór zwany źrenicą (łac. pupilla). Na jej przedniej powierzchni obecne są fałdy tęczówki - nieregularne zagłębienia i uwypuklenia.

Tęczówka zawiera dwa układy włókienek mięśniowych, które działając antagonistycznie sprawiają, że działa ona jak przysłona i reguluje dopływ światła do soczewki. Mięsień zwieracza źrenicy (łac. m. sphincter pupilae), unerwiony przez włókna przywspółczulne, ma wiązki mięśniowe ułożone spiralnie i znajduje się przy brzegu źrenicy. Mięsień rozwieracza źrenicy (łac. m. dilatator pupilae), unerwiony przez włókna współczulne, ma wiązki mięśniowe ułożone promieniście między obwodem tęczówki a źrenicą.

Wyróżnia się cztery warstwy tęczówki:

- nabłonek przedni tęczówki - nabłonek jednowarstwowy płaski
- warstwa graniczna zewnętrzna; brak w niej naczyń krwionośnych
- zrąb tęczówki
- warstwa barwnikowa tzw. część tęczówkowa siatkówki

Analiza tęczówki oka człowieka wykorzystywana jest w irydologii.

Biometryka ta posiada wszystkie Pożądane własności biometryk (wg Clarke'a), czyli:

- Uniwersalność: Każda osoba powinna mieć daną cechę biometryczną.
- Jednoznaczność: Powinna istnieć możliwość rozróżnienia każdych dwóch osób na podstawie danej cechy.
- Trwałość: Niezmienność w czasie.
- Ściągalność: Biometryka powinna być możliwa do zmierzenia w miarę możliwości bezinwazyjnie.
- Akceptowalność: Populacja nie powinna mieć zbyt wielkich obiekcji przeciwko pomiarowi danej biometryki.

Transformata kosinusowa - rodzaj blokowej transformacji danych. Wykorzystuje ona rozwinięcie sygnały w bazie funkcji ortogonalnych zbudowanych z wielomianów Czebyszewa. Macierz przekształcenia kosinusowego jest tworzona na drodze dyskretyzacji wielomianów.

3. Realizacja zadania

W celu realizacji zadania skorzystano z bazy tęczówek dostarczonej przez prowadzącego. Spośród 3 różnych baz wybrano bazę CASIA-IrisV3-Interval. Wybór padł na tę właśnie bazę, ponieważ z pozostałymi dwoma były małe problemy. W przypadku jednej z pozostałych, liczba próbek była zbyt mała, żeby można było zapewnić chociaż w minimalnych stopniu warunku do uczenia się sieci neuronowej. W drugim natomiast przypadku, zdjęcia były prawie 4 krotnie większe, przez co uznałem, że nie ma potrzeby zbyt obciążać komputera i czekać na uzyskanie wyników do kilku razy dłużej (czas dla przeprowadzonych badań i tak był rzędu około 5-10min). Z podanej bazy wybrałem 10 użytkowników. Nie byli to kolejni użytkownicy,

ponieważ nie wszyscy mieli po 8 obrazków tęczówek, dlatego finalnie zdecydowałem się na użytkowników:

```
nazwy_ludzi =
['001';'002';'007';'008';'011';'019';'028';'029';'030';'036';];
```

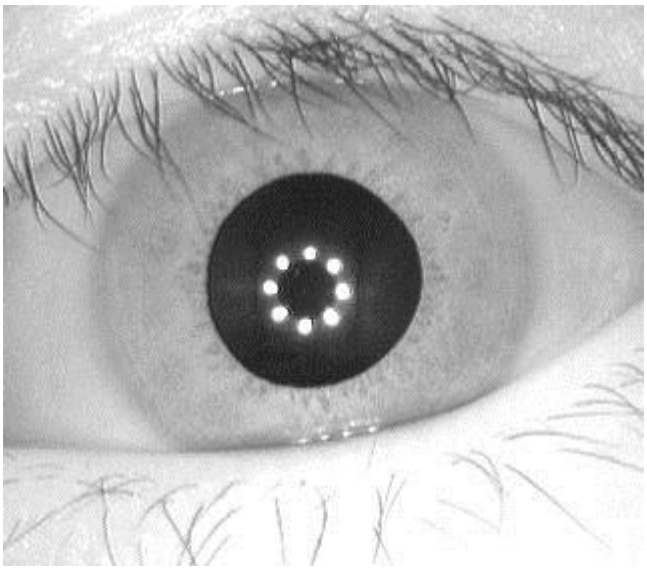
Proces wydobywania tęczówki z obrazka:

W Internecie natrafiłem na bardzo ciekawą metodę znajdowania kółek na obrazkach zaimplementowaną w postaci skryptów programu MATLAB.

Schemat wydobywania tęczówki z obrazka za pomocą znalezionego programu wyglądał następująco:

1. Wgranie obrazka za pomocą funkcji imread()
2. Za pomocą funkcji findcircle() z odpowiednimi parametrami, możliwe było uzyskanie odpowiednich okręgów. Funkcja zwraca współrzędne środka okręgu i promień.

Np. dla zadanego obrazka:



wykonane zostało kilka czynności:

Przede wszystkim dla każdego obrazka przyjąłem założenie, że analiza tęczówki będzie odbywała się w oparciu o dwa okręgi. Pierwszy z nich to okrąg ograniczający tęczówkę, natomiast drugi to okrąg ograniczający źrenicę.

Zastosowana została dwa razy funkcja findcircle().

Za pierwszym razem z następującymi parametrami:

```
[row1, col1, r1]=findcircle(A,100,250,0.4,15,0.1,0.1,1,0.5)
```

Następnie:

```
[row, col, r]=findcircle(A,30,60,0.4,15,0.1,0.10,1,0.5)
```

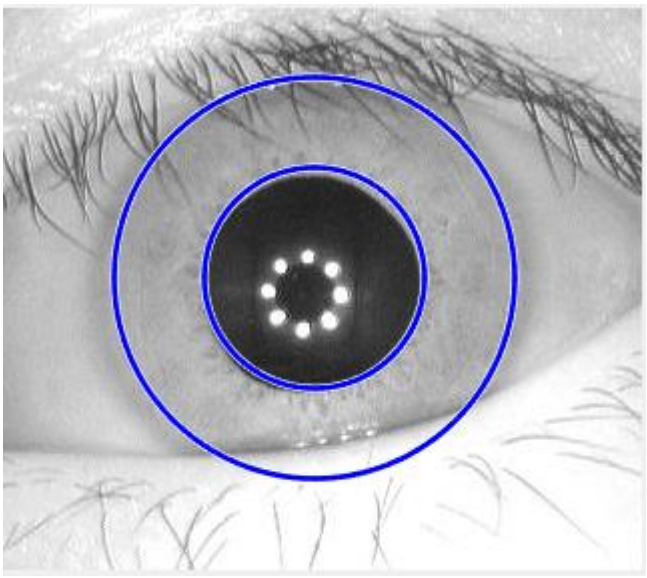
Parametry dla tej funkcji opisane zostały w helpie.

findcircle

```
  findcircle - returns the coordinates of a circle in an image using the
Hough transform
  and Canny edge detection to create the edge map.

  Usage:
  [row, col, r] = findcircle(image,lradius,uradius,scaling, sigma, hithres,
lowthres, vert, horz)

  Arguments:
        image              - the image in which to find circles
        lradius            - lower radius to search for
        uradius            - upper radius to search for
        scaling            - scaling factor for speeding up the
                                 Hough transform
        sigma              - amount of Gaussian smoothing to
                                 apply for creating edge map.
        hithres            - threshold for creating edge map
        lowthres           - threshold for connected edges
        vert               - vertical edge contribution (0-1)
        horz               - horizontal edge contribution (0-1)

  Output:
        circleiris         - centre coordinates and radius
                                 of the detected iris boundary
        circlepupil        - centre coordinates and radius
                                 of the detected pupil boundary
        imagewithnoise  - original eye image, but with
                                 location of noise marked with
                                 NaN values

  Author:
  Libor Masek
  masekl01@csse.uwa.edu.au
  School of Computer Science & Software Engineering
  The University of Western Australia
  November 2003
```

Jak widzimy, funkcja przyjmuje na początku ścieżkę do obrazka, którego analizy ma dokonać, następnie dolną i górną granicę wielkości promienia okręgu. W naszym przypadku po analizie rozmiarów obrazka w obrazie Paint, przyjąłem założenie, ze rozmiar źrenicy dla każdego obrazka waha się w granicach 30 – 60 pxl, natomiast dla tęczówki w granicach 100 – 250 pxl. Dobranie takich parametrów gwarantuje, że dla jednego obrazka nie zostaną dwa razy wzięte te same okręgi.
Jeżeli chodzi o kolejne parametry, to zostały one dobrane w drodze doświadczeń i przeprowadzania serii testów. Dla parametru scaling, wziąłem dość małą wartość (0.4) ponieważ dla większej, moce obliczeniowe mojego komputera okazywały się niewystarczające. Następnym ciekawym parametrem jest hithres oraz lowthress. Są to odpowiednio górne i dolne progi dla których dokonywana jest detekcja brzegu. Przy

większej wartości granicznej, aby funkcja mogła znaleźć odpowiedni okrąg, to sąsiadujące ze sobą pixele musiałyby mieć dość duży kontrast. W przypadku naszych zdjęć, niestety nie mamy dość dużego kontrastu i jedynie możemy rozróżniać obiekty na zdjęciu w kategorii „jasny" i „jaśniejszy". Dlatego tez wartości tych współczynników ustawione są na 0.1. Dla zadanego powyżej przykładowego obrazka, otrzymaliśmy następujące rezultaty:



Jak widzimy, poprawnie znaleziono dwa okręgi dobrze ograniczające szukany obszar. Niemniej jednak w celu jeszcze lepszego dopasowania wykorzystano kilka narzędzi aparatu matematycznego.

Przede wszystkim, należy zauważyć że dane okręgi nie mają wspólnego środka, dlatego ekstrakcja od jednego punktu uznanego za środek okręgu byłaby krzywdząca dla któregoś z okręgów. W celu polepszenia tej sytuacji, ze środków okręgów wyciągnięto średnią, przez co otrzymano współrzędne punku będącego pomiędzy dwoma środkami.

```
srodek=[(col+col1)/2 (row+row1)/2]
```

Następnie wprowadzono zmienną

```
srednie_r=(r1-15)
```

jest to wartość promienia większego okręgu pomniejszona o 15. Zabieg ten został zastosowany w celu usunięcia dolnej i górnej powieki, oraz co większych fragmentów brwi.

Następnym etapem podczas realizacji zadania było przetransformowanie wartości pixeli do wektora. W tym celu, utworzono wektor K, oraz utworzono pętlę:

```
for n=1:sizex
            for l=1:sizey
                if(((n-srodek(1))^2+(l-srodek(2))^2)<=srednie_r^2)
                    if(((n-srodek(1))^2+(l-srodek(2))^2)>r^2)
                        K=horzcat(K,A(n,l));
                    end
                end
            end
        end
```

W pętli tej, przechodzimy po każdym pixelu z wyjściowego obrazka i dla niego sprawdzamy dwa warunki. Jeśli punkt należy do źrenicy, czyli jego współrzędne są oddalone od środka średniego o mniej niż r1-15 a więcej niż r (usunięcie źrenicy), to wartość pixela dodawana jest to wektora. Usunięcie źrenicy ma na celu polepszenie otrzymanych wyników. W przypadku, gdy nie usunęlibyśmy źrenicy, mogłoby się okazać, że wszystkie próbki są bardzo do siebie podobne, a przynajmniej wartości transformat w znacznej mierze się pokrywają, przez co identyfikacja byłaby dużo trudniejsza.

W dalszej części realizacji zadania skorzystano z poprzedniego zadania laboratoryjnego dotyczącego odcisków palców. Podobnie do tamtego zadania, tak i tutaj, dla każdego wektora otrzymanego sposobem podanym u góry, dokonano jego transformaty kosinusowej, a następnie za pomocą dostarczonego przez prowadzącego modułu zigzag obliczono wektor_po_zigzagu. Następnie podzielono otrzymany wektor na kilka podziałów, i dla każdego fragmentu policzono średnią. Dla podziału 180 częściowego otrzymano przykładowy wektor:

```
>> PP(:,1)'
ans =
  Columns 1 through 20
  362.8601    9.1597  156.0155  124.7669  141.4297   84.5777   74.9120   87.3440   92.4304   97.6715   36.1021   36.5978   45.9858   56.9447   48.6421   23.6428   18.6681   25.6530   41.3716   28.9581
  Columns 21 through 40
   15.3538   13.5229   18.7842   35.4779   15.1852   16.3268   14.9416   16.6602   29.3698   17.0279   14.9758   13.4130   17.6063   20.7853   12.9344   13.8556   15.0833   18.5366   12.3470    8.9856
  Columns 41 through 60
   10.2167   16.2722   14.9025    8.8098    9.0569   10.8203   14.2860   12.8926   10.0193    8.3287    9.5550   11.5575   11.1148    8.8627    8.3225    9.5598   10.5272    9.6941    7.8471    7.3516
  Columns 61 through 80
    9.3734   10.2105    6.9311    8.4142    8.4169    9.1635    8.9322    5.4017    8.3171    7.2415    7.8990    7.5418    6.4030    6.5848    7.4938    7.9868    7.3305    6.1677    7.0400    6.9359
  Columns 81 through 100
    9.1725    4.9746    4.6915    6.0739    6.2336    8.4995    6.0365    5.1484    5.6663    6.2863    6.4653    5.2523    6.4560    5.5554    7.1850    5.6698    4.9837    6.4481    5.0297    6.4749
  Columns 101 through 120
    5.1957    4.9277    5.2796    5.0694    5.7797    4.9084    4.6649    4.6258    5.4965    5.8871    4.6435    4.8036    4.2494    6.8132    4.9690    5.0698    4.2461    4.7236    5.7450    4.8726
  Columns 121 through 140
    4.6267    3.7349    4.8159    5.3970    4.0562    4.3544    4.4140    5.9990    4.3208    4.0523    4.0631    4.2385    5.7691    4.7880    4.0554    4.0300    3.9863    5.6868    4.0051    3.8082
  Columns 141 through 160
    4.4913    3.9743    4.9385    3.8050    3.3726    4.3263    5.0236    3.7539    4.2291    3.6156    3.4531    5.4972    3.6716    4.0011    4.0185    3.2829    4.7646    3.6451    3.4491    3.2283
  Columns 161 through 180
    3.8341    4.2416    3.1607    3.7089    2.9065    4.4948    3.0944    3.4912    3.2658    3.3309    4.1968    3.7183    3.0335    3.5008    3.4659    4.5385    3.2766    3.0577    3.6091    3.8937
```

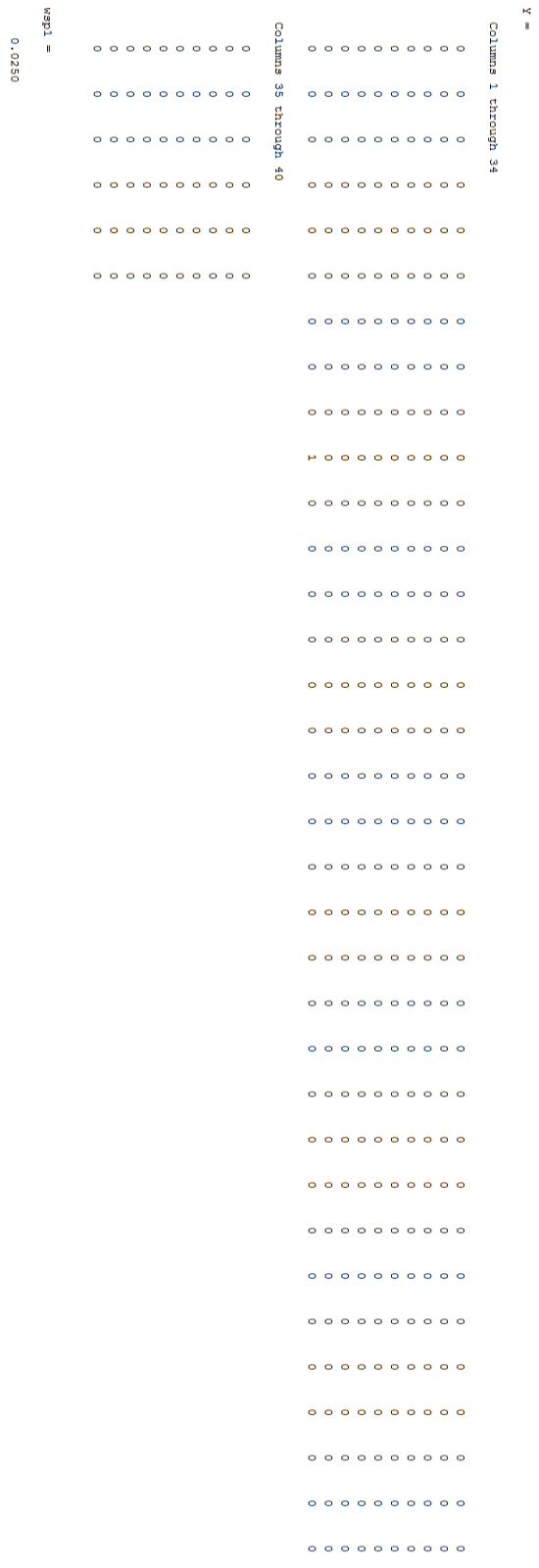
Następnie utworzono 40 takich wektorów (10 użytkowników po 4 obrazy tęczówki dla zbioru uczącego się) oraz skonstruowano 3 warstwową sieć neuronową. Sposób wyboru sieci został dokładniej omówiony podczas realizacji poprzednich zadań.

Przetestowano wyjściowy zbiór, otrzymując współczynnik identyfikacji = 1, co oznacza, że wszystkie próbki, które zostały wykorzystane do nauki, zostały następnie dobrze zidentyfikowane.

Kolejnym krokiem było sprawdzenie współczynnika identyfikacji dla kolejnych 4 już innych próbek dla każdego użytkownika.

W tym celu powtórzono całą operacją. Wyniki otrzymane prezentują się następująco:

a) dla ilości podziałów = 16 otrzymano bardzo mały współczynnik na poziomie 0.025.



b) dla ilości podziałów = 64 otrzymano wyniki 0.275

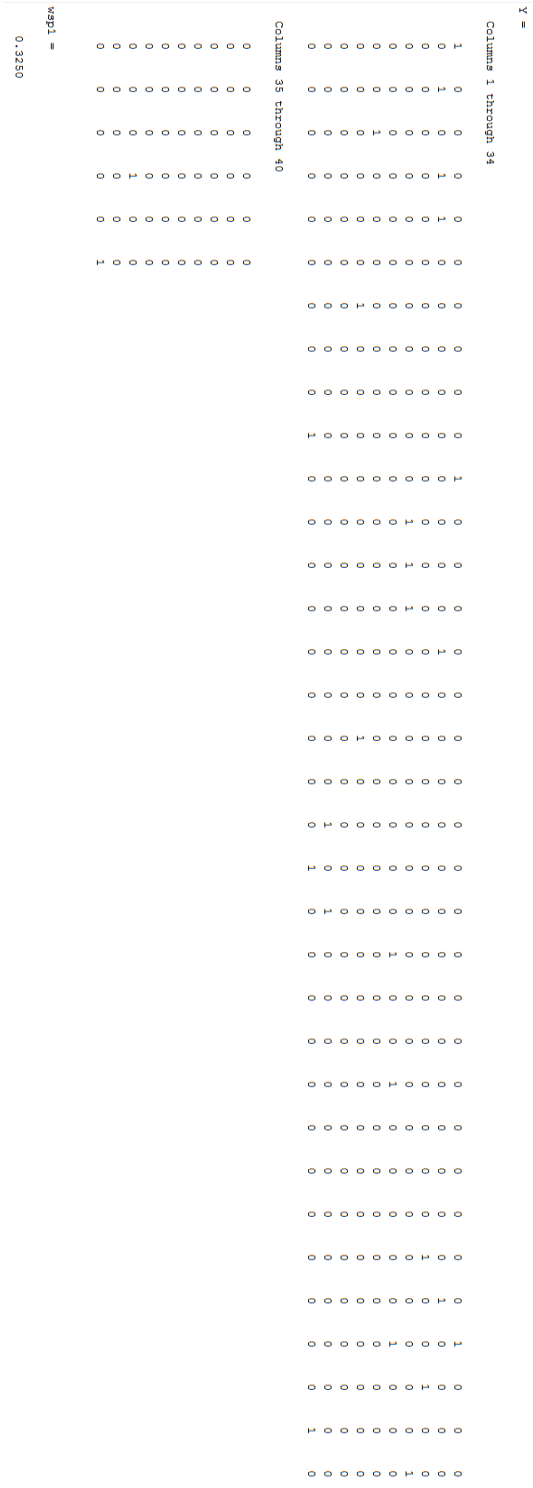
```
Y =

  Columns 1 through 34

 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 1
 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0

  Columns 35 through 40

 0 0 0 0 0 0
 0 0 0 0 0 0
 0 0 0 0 0 0
 1 0 0 0 0 0
 0 0 0 0 0 0
 0 0 0 1 0 0
 0 0 0 0 0 0
 0 0 0 0 0 0
 0 1 0 0 0 0
 0 0 0 0 0 1

wsp1 =

    0.2750
```

c) Dla ilości podziałów = 180 otrzymano współczynnik 0.25. Warto w tym miejscu zauważyć, że mimo wzrostu ilości podziałów, wartość współczynnika nie zwiększa się. Dla przeprowadzonych przeze mnie badań okazało się, że najlepsza liczba podziałów to 64.

```
Y =

  Columns 1 through 34

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 1 1 0 1 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

  Columns 35 through 40

0 0 0 0 0 0
1 0 0 0 0 1
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 1 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 1 0 0 0 0

wsp1 =

    0.2500
```

d) Co można zrobić, żeby jeszcze poprawić sytuację?

Skoro nie pomaga zwiększanie liczby przedziałów, jedynym rozwiązaniem wydaje się wpłynięcie na sieć neuronową. Analiza wyników dla jej współczynników wykazała, że dla parametru goal 1000 krotnie mniejszego niż poprzednio, udało się otrzymać dość znaczącą poprawę wyników. Jest ona teraz na poziomie 0.325. Niestety zmniejszanie parametrów w nieskończoność nie prowadzi do niczego dobrego. Wręcz przeciwnie, sieć staje się przetrenowana i umie reagować tylko na wyjściowe próbki.



e) wsp1 odpowiada wartości poprawnie zidentyfikowanych próbek z nowo wybranego zbioru natomiast wartość wsp odpowiada tej samej wartość tylko że dla zbioru, który był użyty dla uczenia się. Jak widzimy poprawność rzędu 1 pozwala stwierdzić, że sieć neuronowa poprawnie się nauczyła.

```
wsp1 =

    0.2250

>> wsp

wsp =

     1
```

4. Wnioski

Większość wniosków została przedstawiona w rozdziale pt. „Realizacja zadania”. Należy dodać, że największy wpływ na wyniki mają ilość warstw użytych dla sieci neuronowej oraz liczba podziałów wektora_po_zigzagu. Dobierając odpowiednie współczynniki można uzyskać ciekawe rezultaty, natomiast trzeba pamiętać, że aby wyznaczyć optymalne parametry nie należy kierować się zasadą, że im więcej tym lepiej.

Podsumowując, realizację zadania projektowego oceniam bardzo dobrze. Zapoznałem się ze wszystkimi informacjami, które pomogły mi w zbudowaniu dość dobrze działającego systemu identyfikującego użytkowników za pomocą tęczówki. Ponadto rozwinąłem umiejętność operowania w środowisku Matlab oraz poszerzyłem swoją wiedzę matematyczną.