AKADEMIA GÓRNICZO-HUTNICZA

im. Stanisława Staszica w Krakowie

WYDZIAŁ INŻYNIERII MECHANICZNEJ I ROBOTYKI

Projekt inteligentnego systemu do rozpoznawania odcisków palców

Mateusz Danieluk

Imię i nazwisko

Mechatronika

Kierunek studiów

Systemy Inteligentne

Przedmiot

Kraków, rok 2015/2016

Spis treści

Wstęp

Odcisk palca pozwala na jednoznaczne przyporządkowanie go do określonej osoby, ponieważ każdy człowiek posiada własny, unikalny rozkład linii papilarnych odróżniających go od całej populacji. W dzisiejszych czasach, gdzie istotną rolę odgrywa bezpieczeństwo i tworzone są coraz to nowsze systemy zabezpieczeń wykorzystanie linii papilarnych do identyfikacji odgrywa znaczącą rolę. Zalet zabezpieczenia odciskami palców jest wiele. Dwie główne to unikalność (każdy układ linii papilarnych jest inny) oraz "kompaktowość" (do identyfikacji nie jest wymagany żaden dodatkowy przedmiot, wystarczy odcisk).

Niniejsza praca przedstawia projekt systemu do rozpoznawania odcisków palców. Jest to kolejny projekt, dotyczący systemów inteligentnego budynku. Taki system mógłby z powodzeniem znaleźć zastosowanie do otwierania drzwi.

W projekcie użyto programu MATLAB, wykorzystując przy tym możliwość stworzenia sieci neuronowej.

Pierwszy rozdział projektu przedstawia cel i założenia projektu, gdzie przedstawiono zakres prac i szczegóły dotyczące założeń. W rozdziale drugim prezentowane są najważniejsze zagadnienia związane z odciskami palców i sieciami neuronowymi. Kolejny rozdział dotyczy etapu przygotowania próbek, w którym prezentowane są szczegóły dotyczące pozyskiwania odcisków i informacje o ich obróbce graficznej w pakiecie Matlab. Rozdział czwarty to analiza w programie MATLAB, gdzie zawarto najważniejsze fragmenty kodu. Rozdział piąty prezentuje wyniki uczenia sieci neuronowej. Ostatni rozdział to wnioski, które można sformułować na podstawie wyników.

1. Cel i założenia projektu

1.1. Cel projektu

Celem projektu jest stworzenie programu w oprogramowaniu Matlab do rozpoznawania odcisków palców. Próbki należy zarejestrować samodzielnie, mają to być odciski własnych 5 palców.

Należy pobrać 10 próbek odcisków każdego z 5 palców. Następnie w środowisku Matlab napisać program który przeprowadzi obróbkę obrazu, umożliwiającą wyznaczenie parametrów, będących wektorem wejściowym do sieci neuronowej. Sieć ta ma umożliwiać rozpoznanie każdego palca.

1.2. Założenia projektu

Próbki należy pobrać samodzielnie, mają to być odciski z jednej dłoni z każdego palca. W tym celu można wykorzystać czytnik linii papilarnych lub też tusz. Macierz ucząca będzie składać się z parametrów dla 7 próbek z każdego palca, natomiast macierz testująca będzie miała po 3 próbki. Sieć będzie rozpoznawać palce spośród 5 różnych. Skuteczność algorytmu powinna przekraczać 90%.

2. Wprowadzenie teoretyczne

2.1. Odciski palców

Linie papilarne, to charakterystyczny układ bruzd na skórze ssaków naczelnych, w szczególności na opuszkach palców rąk, ale również na wewnętrznej powierzchni dłoni, palcach stóp i wargach, a u niektórych gatunków na spodniej stronie ogona. Poza naczelnymi, obecność linii papilarnych stwierdzono u koali.

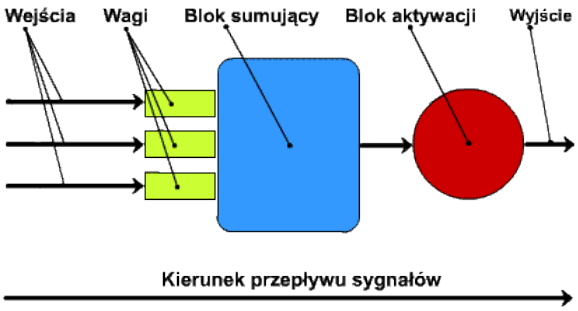
Linie papilarne opisuje się za pomocą tzw. minucji – charakterystycznych cech, takich jak początki, zakończenia, rozwidlenia, haczyki itp. Wzajemny układ minucji jednoznacznie identyfikuje daną osobę. Do uznania dwóch śladów linii papilarnych za tożsame wystarczy od kilku do kilkunastu cech wspólnych (ta sama minucja występująca w tym samym miejscu). Przyjmuje się, że 12 cech wspólnych wystarcza do pewnego określenia tożsamości, choć niektóre minucje, występujące rzadziej lub specyficzne dla danej populacji, zmniejszają tę liczbę. Wszystkie typy minucji są kombinacją dwóch podstawowych elementów: rozwidleń oraz zakończeń grzbietów.

Typy minucji zaprezentowano na rysunku 2.1.

Rys.2.1. Typy minucji
(http://sequoia.ict.pwr.wroc.pl/)

2.2. Sieci neuronowe

Sztuczne sieci neuronowe to modele obliczeniowe powstałe z interdyscyplinarnej synergii klasycznych dziedzin nauki, w skład których wchodzą: biologia, matematyka, fizyka. Ich budowa oraz schemat przetwarzania informacji został zaprojektowany zgodnie ze strukturami tworzącymi ludzki system nerwowy. Podstawowym elementem strukturalnym jest, tak jak w naturze, neuron.



Rys.2.2. Budowa sztucznego neuronu
(www.cs.put.poznan.pl)

Sztuczny neuron to prosty element, który odpowiada za przetwarzanie sygnałów w matematycznych modelach sieci. Skonstruowany został na wzór neuronu naturalnego. Wejścia to odpowiedniki dendrytów, lub ściślej: sygnały przez nie nadchodzące. Wagi to cyfrowe odpowiedniki modyfikacji dokonywanych na sygnałach przez synapsy. Blok sumujący to odpowiednik jądra, blok aktywacji to wzgórek aksonu, a wyjście - to akson.

Pojedyncze neurony są łączone w struktury, czyli sieci.Podstawowym zadaniem sztucznych sieci neuronowych w ogólnym przypadku jest wnioskowanie funkcji na podstawie obserwacji. Istotnym pojęciem w przypadku sieci neuronowych jest ich uczenie. Pod pojęciem uczenia sieci rozumiemy wymuszenie na niej określonej reakcji na zadane sygnały wejściowe. Sieci neuronowe wykazują tolerancje na nieciągłości, zakłócenia czy też nawet

braki w zbiorze uczącym. Dzięki temu mogą być stosowane tam, gdzie nie efektywne bądź niemożliwe jest rozwiązanie problemu w inny sposób.

Przykładowe obszary zastosowania sieci neuronowych:

- diagnostyka medyczna;
- prognozowanie zmian cen rynkowych,
- rozpoznawanie wzorców (sygnałów mowy, symboli, pisma, odcisków, itp.)
- modelowanie procesów biznesowych,
- aproksymacja wartości funkcji.

Strukturami wykorzystanymi do realizacji celu tego projektu będzie perceptron oraz feedforward backpropagation network, czyli sieć wielowarstwowa jednokierunkowa. Ta pierwsza jest liniowym algorytmem klasyfikującym, której wyjściem $f(x)$ jest wartość binarna.

$$\mathrm{f(x)} = \begin{cases} 1, & \text{jeżeli } \mathrm{w} * \mathrm{x} + \mathrm{b} > 0 \\ 0 & \end{cases}$$

Gdzie w – wektor wag, b – bias (wartość stała określana na początku nauki). Proces nauki polega na takim określeniu wartości wag dla wszystkich wartości wejściowych parametrów, aby granica decyzyjna dana wzorem $\sum_{i=1}^{n} w_i x_i + b = 0$ (n-1 wymiarowa płaszczyzna) odpowiednio klasyfikowała próbki. Jest to proces uczenia z nauczycielem, gdzie oprócz wartości sygnałów (macierz ucząca) podaje się wartości docelowe tych sygnałów (macierz celu).

3. Przygotowanie próbek i ich obróbka

3.1. Przygotowanie próbek

Pierwszym etapem projektu było pobranie odcisków palców. W tym celu wykorzystano tusz do stempli. Na palec była nanoszona niewielka warstwa tuszu, a następnie odcisk nanoszone na kartkę. Z każdego palca pobrano około 30 próbek, aby po zeskanowaniu kartki można było wybrać 10 najlepszych. Etap ten wymagał wielu prób. Próbowano maczać palec bezpośrednio w tuszu, a także używać do tego celu nasiąkniętej tuszem gąbki. W obydwu przypadkach rezultaty były niezadowalające. Najlepsze rezultaty dała specjalna poduszeczka do stempli nasiąknięta tuszem.

Poniżej przedstawiono po jednej próbce dla każdego palca:

Rys.3.1. Odcisk kciuka

Rys.3.2. Odcisk palca wskazującego

Rys.3.3. Odcisk palca środkowego

Rys.3.4. Odcisk palca serdecznego

Rys.3.5. Odcisk małego palca

3.2. Obróbka graficzna

Aby otrzymać jak najlepsze wyniki, obrazy zostały poddane szeregowi zabiegów. Miały one na celu pozostawienia na obrazie wąskich linii obrazujących przebieg linii papilarnych wraz z pozbyciem się zakłóceń, takich jak pojedyncze piksele, przerwy w linii, czy rozgałęzienia. Przebieg obróbki:

- Binaryzacja :
  binarny = im2bw(obraz, wsp_t);

Rys.3.6. Obraz po binaryzacji

- Odwrócenie kolorów:
  odwrocony = imcomplement(binarny);

Rys.3.7. Obraz po odwróceniu kolorów

- Otwarcie (erozja obcina brzegi obiektu na obrazie, dylatacja zamyka małe otwory)
  otwarcie1 = imopen(inversed, se);

Rys.3.7. Obraz po otwarciu

- Zamknięcie (dylatacja, a następnie erozja)
  zamkniecie1 = imclose(otwarcie1, se);

Rys.3.7. Obraz po zamknięciu

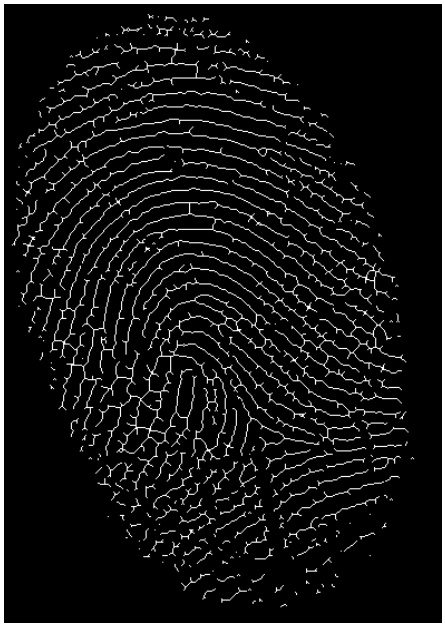
- Filtr medianowy (pozbycie się szumów, wsp_medfilt uzależniony od palca)

  m = medfilt2(zamkniecie1, [wsp_medfilt wsp_medfilt]);

Rys.3.8. Obraz po operacji z użyciem filtru medianowego

- Szkieletyzacja

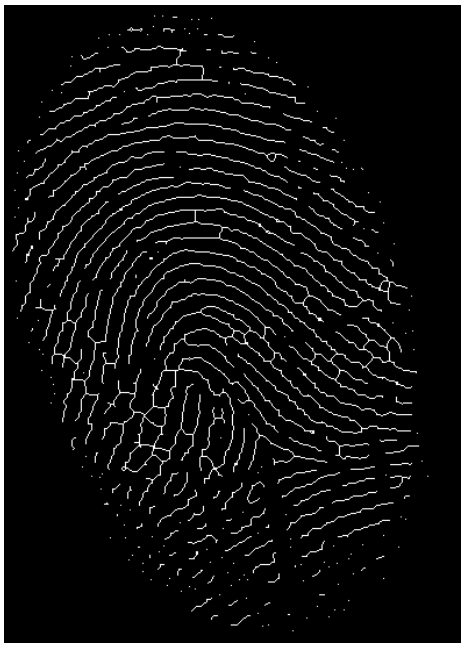
  szkielet = bwmorph(m, 'skel', inf);



Rys.3.9. Obraz po szkieletyzacji

- Usunięcie/obcięcie gałązek

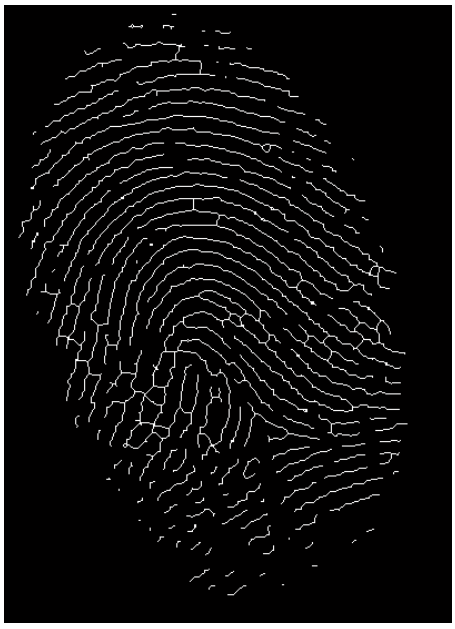
  galazki = bwmorph(szkielet, 'spur', 4);



Rys.3.10. Obraz po usunięciu gałązek

- Usunięcie pojedynczych pikseli (szumów)

  pojedyncze = bwmorph(galazki, 'clean',4);



Rys.3.11. Obraz po usunięciu pojedynczych piksel

Kolejnym krokiem jest wyznaczenie minucji. W prezentowanym projekcie wyznaczane będą tylko rozwidlenia i oczka – reprezentują one wystarczającą informację o kształcie linii papilarnych i są stosunkowo łatwe do znalezienia.

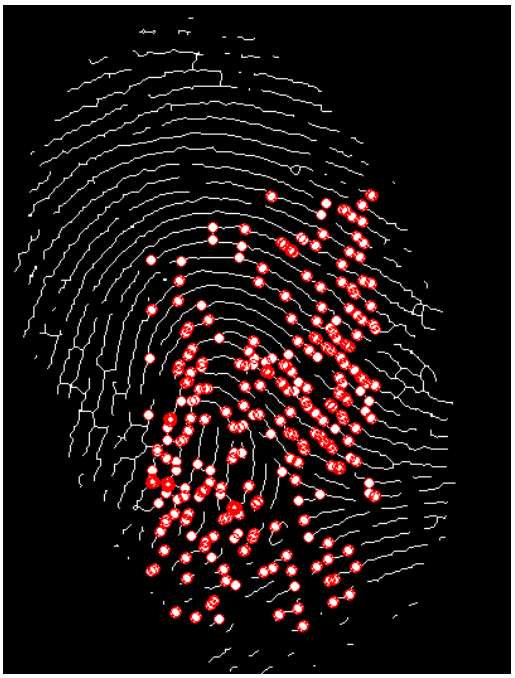
Wyznaczanie minucji polega na znalezieniu miejsc w których szkielet obrazu po szkieletyzacji się rozgałęzia. Istnieje zaimplementowana funkcja do tego celu:

```
minucje = bwmorph(pojedyncze,'branchpoints');
```

Aby usunąć minucje pojawiające się tuż przy krawędziach zaimplementowano algorytm, który bierze pod uwagę jedynie minucje w ustalonej odległości od krawędzi poziomej i pionowej w zależności od palca (określa to zmienna krawedz_x i krawedz_y):

```
nr=1;

        for k=krawedz_y:1:length(minucje(:,1))-krawedz_y
           for s=krawedz_x:1:length(minucje(1,:))-krawedz_x
              if ( (minucje(k,s) == 1)  )
                 x_minucje(nr,idx)=s;
                 y_minucje(nr,idx)=k;
                 nr=nr+1;
              else
              end
           end
        end
```



Rys.3.12. Obraz z minucjami wyznaczonymi w określonym obszarze

4. Identyfikacja odcisków

4.1. Wybór parametrów do macierzy uczącej

Aby możliwa była identyfikacja próbek przez sieć neuronową konieczne było stworzenie macierzy uczącej zawierające charakterystyczne parametry dla każdego odcisku. Po wielu próbach dobrano następujące wartości:

```
% ilość znalezionych minucji na obrazie
ilosc_minucji = length(y_minucje(:,idx));

% średnia wartość współrzędniej Y z wszystkich minucji
y_mean_minucji = mean(y_minucje(:,idx));

% średnia wartość współrzędniej X z wszystkich minucji
x_mean_minucji = mean(x_minucje(:,idx));

% wariancja współrzędnej X Minucji
x_minucji_var = var(x_minucje(:,idx));

% odchylenie standardowe współrzędnej X minucji
x_minucji_std = std(x_minucje(:,idx));

% wariancja współrzędnej Y Minucji
y_minucji_var = var(y_minucje(:,idx))

% odchylenie standardowe współrzędnej Y minucji
y_minucji_std = std(y_minucje(:,idx));

% średnia odległość od pierwszzej minucji
sr_odl_od_1_M = mean(odleglosci(:,idx));

% wariancja zbioru odległości od pierwszej minucji
var_odl_od_1_M = var(odleglosci(:,idx));

% odchylenie standardowe zbioru odległości od pierwszej minucji
odch_odl_od_1_M = std(odleglosci(:,idx));
```

Dodatkowo dobrano położenie 30 minucji jako dane do macierzy uczącej:

```
x_minucje(1:ile_minucji,idx);
y_minucje(1:ile_minucji,idx)
```

Po tym etapie dokonano normalizacji wielkości z macierzy uczącej i przystąpiono do nauczania.

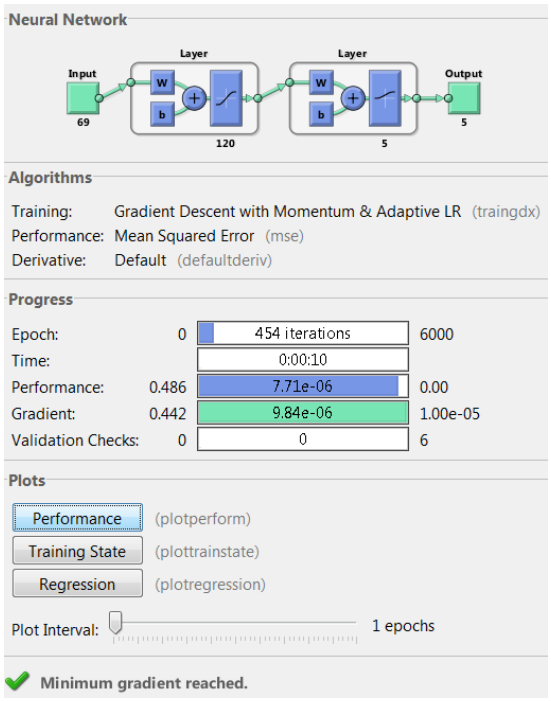
4.2. Nauka sieci neuronowej

Do nauki sieci neuronowej przeznaczono 7 próbek uczących z każdego palca (łącznie 35 próbek) oraz 3 testujące (łącznie 15). Dodatkowo należało stworzyć macierz celu, która określała przynależność danego odcisku do danego palca:

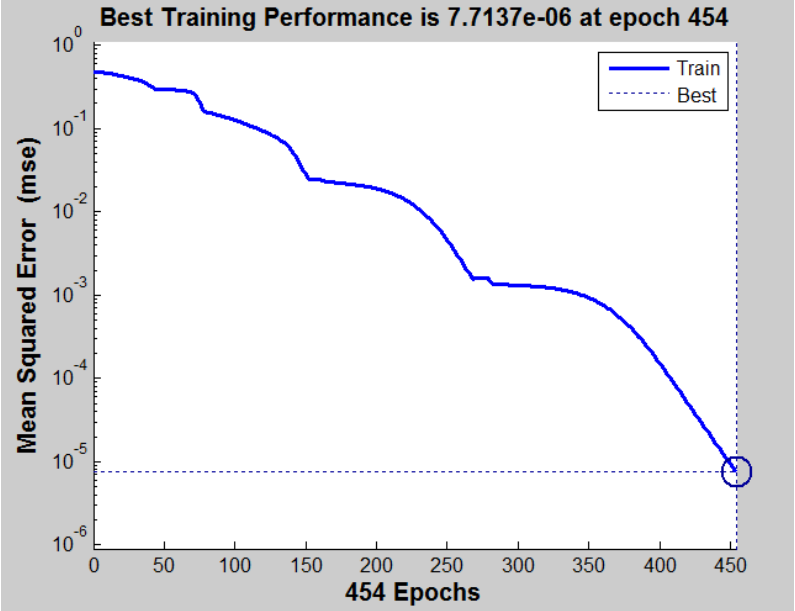
```
M_cel = zeros(4, length(M_ucz(1, :)));
  kolejnosc= 1;
 for j=1:1:length(M_ucz(1, :))
        M_cel(kolejnosc,j)=1;
        if mod(j,liczba_probek_uczacych)==0
           kolejnosc=kolejnosc+1;
        end
  end
```

Do realizacji zadania rozpoznawania odcisków palców wybrano trójwarstwową sieć neuronową (jedna warstwa ukryta, jedna wejściowa, jedna wyjściowa). Sieć ta ma 69 neuronów w warstwie wejściowej, 120 w warstwie ukrytej i 5 w warstwie wyjściowej (5 rozróżnianych odcisków palców).

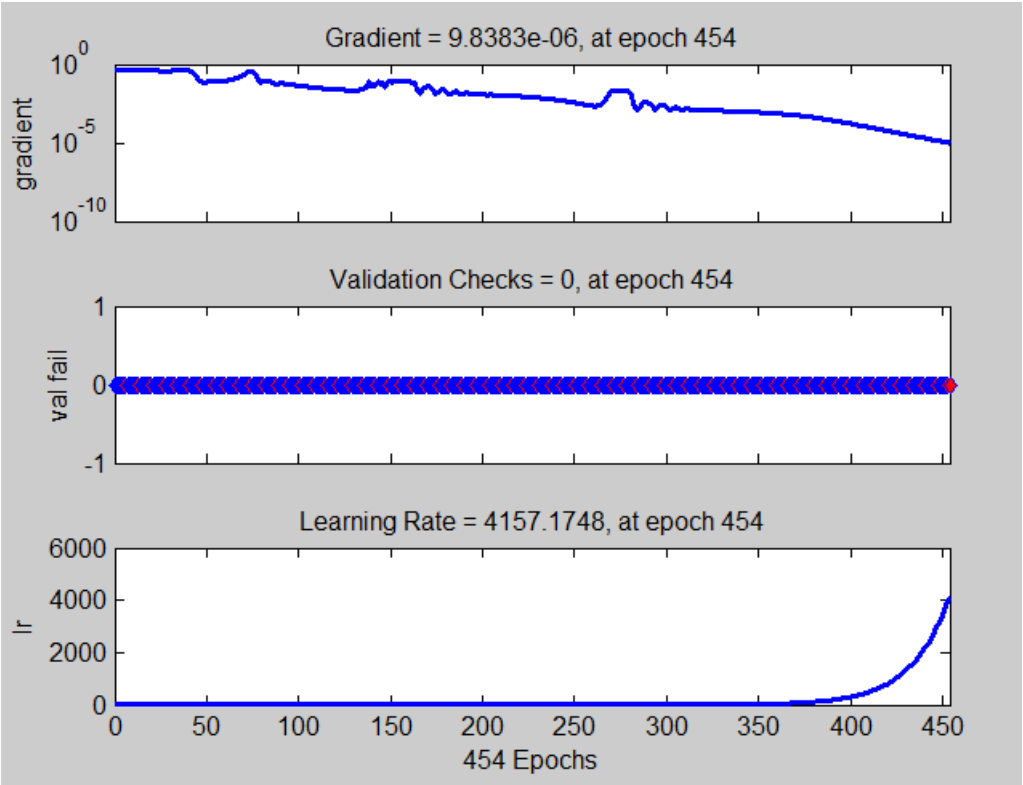
```
tic
   B=[0 1];
   PR=repmat(B,69,1); % 69 parametrów wejściowych
   mnet=newff(PR,[120 5],{'tansig' 'logsig'},'traingdx'); % 120 neuronów w warstwie ukrytej i 5
w warstwie wyjęciowej
   mnet=init(mnet);
   mnet.trainParam.show=10000;
   mnet.trainParam.epochs=6000;
   mnet.trainParam.goal=0;
   mnet=train(mnet,M_ucz,M_cel);
   toc
   Pegz=M_test;
   YY=sim(mnet,Pegz);
   yyt = YY';
```



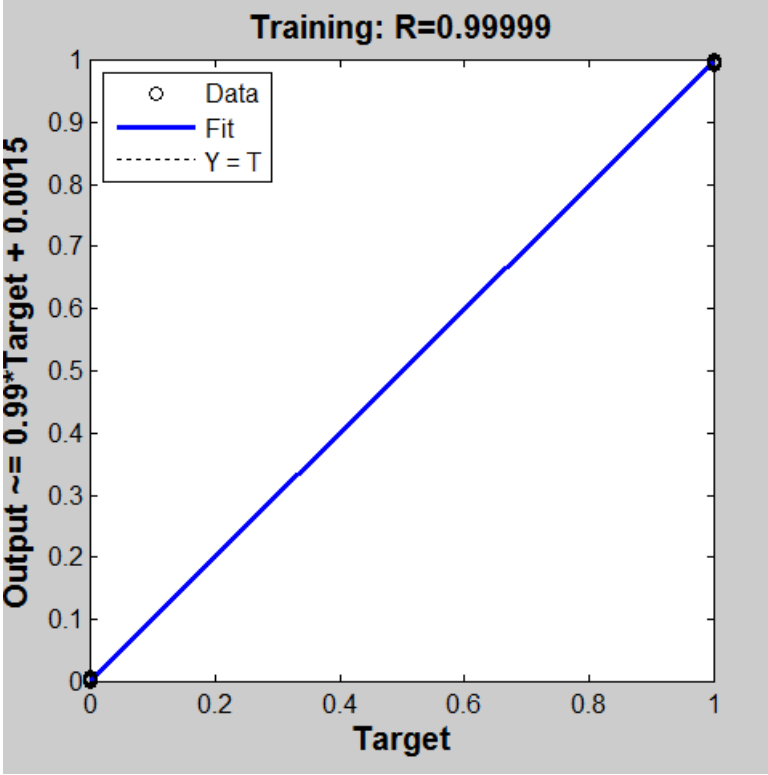
Rys.3.13. Panel sieci neuronowej



Rys.3.14. Przebieg procesu uczenia



Rys.3.14. Panel procesu uczenia



Rys.3.15. Regresja

5. Wyniki

Testowanie sieci neuronowej obejmowało 3 próbki dla każdego palca. Każda "trójka" musiała zostać odpowiednio przydzielona do jednego z pięciu palców. W poniższej tabeli wiersze oznaczają numer palca, a kolumny kolejne próbki testujące.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0.9820	0.9716	0.9820	5.172...	4.824...	5.936...	4.175...	0.0122	3.320...	1.138...	9.064...	0.0020	4.946...	5.598...	7.434...
2	0.0147	0.0044	1.983...	0.9756	0.9970	0.9050	0.0035	3.564...	4.758...	0.0040	0.1621	0.0037	0.0037	0.0318	0.0020
3	3.410...	0.9598	0.7532	3.563...	0.0026	2.061...	0.0201	0.9819	0.0048	1.199...	0.2560	0.0034	5.940...	2.987...	0.0028
4	0.0023	0.0067	0.4052	0.0039	1.457...	0.3221	0.0651	1.041...	0.9813	0.4514	0.3260	0.9937	0.0050	0.0207	8.069...
5	2.487...	2.259...	8.263...	7.919...	7.323...	0.0117	0.0103	0.1236	0.0527	0.1868	1.278...	1.048...	0.9920	0.9858	0.9959

Rys.3.16. Odpowiedź sieci dla moich próbek

Jak widać na powyższej tabeli, zdecydowana większość odcisków zostało prawidłowo przyporządkowana do palca.. Wśród 15 testowanych próbek 11/15 uzyskało pewność rozpoznania większą od 97%, dwie z rozpoznaniem na poziomie 40%, a dwie z praktycznie zerowym rozpoznaniem. Wszystko było spowodowane amatorsko pobranymi próbkami. Aby upewnić się, że sieć działa poprawnie przeprowadzono nauczanie dla próbek, które zostały znalezione w internetowej bazie danych. Próbki te były znacznie lepszej jakości.

Rys.3.17. Przykładowy odcisk znaleziony w internecie

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0.9989	0.9869	0.8253	2.231...	7.776...	1.467...	2.129...	4.770...	1.577...	2.029...	0.0042	0.0047	5.342...	4.242...	0.0031
2	1.462...	3.538...	9.900...	0.9965	0.9947	0.9937	0.0067	2.757...	5.848...	4.642...	1.854...	0.1661	0.0045	5.652...	0.0013
3	8.019...	4.366...	5.593...	0.0076	0.0060	0.0010	0.9929	0.8881	1.0000	9.120...	6.877...	1.986...	2.208...	0.0759	4.289...
4	0.0035	0.2371	2.750...	2.216...	6.746...	6.520...	0.0016	0.1625	3.339...	0.9973	0.9998	0.2489	0.0029	0.0109	9.131...
5	1.359...	3.463...	0.0850	1.738...	2.208...	0.0031	4.472...	5.350...	0.0023	0.0048	0.0092	0.0012	0.9938	0.9459	0.9973

Rys.3.18. Odpowiedź sieci dla próbek znalezionych w internecie.

W przypadku próbek o znacznie lepszej jakości uzyskano 14/15 wyników rzędu 90-100%. Potwierdza to poprawność działania zaimplementowanego algorytmu.

6.Podsumowanie

Przeprowadzona analiza rozpoznawania odcisków linii papilarnych z wykorzystaniem sieci neuronowych wykazała wysoką skuteczność działania i dokładność rozpoznawania.

Istotną kwestią w poprawnym działaniu programu odgrywa dobór próbek. Aby możliwe było prawidłowe rozpoznawanie odcisków palców, dużą wagę należy przywiązać do ich etapu pobierania. Zastosowanie czytnika linii papilarnych zamiast wykorzystania tuszu, który nieraz nierównomiernie rozkłada się na powierzchni paca, zapewne dałoby znacznie lepsze rezultaty.

Co więcej, na uzyskane wyniki wpływ miało także przetwarzanie obrazów próbek. Dobranie odpowiedniego progu binaryzacji oraz operacji morfologicznych okazało się również trudnym zadaniem i wymagało osobnych współczynników dla każdego z pięciu palców. Parametry dobrane w kodzie są najlepsze jakie udało się uzyskać metodą prób i błędów.

Aby upewnić się, że algorytm jest poprawny skorzystano z próbek odcisków o znaczniej lepszej jakości znalezionych w internecie. Poprawne wykrycie 14 z 15 próbek testujących pozwala stwierdzić, iż program działa poprawnie, a dobrane parametry do macierzy uczącej są optymalne. W przypadku własnych odcisków program rozpoznał 11 z 15 próbek, co jest dobrym wynikiem jak na odciski pobrane w warunkach domowych.

Załączniki

Kod programu:

```
clc;
clear all;
close all;

 x_minucje(1,1:50) = 1;
 y_minucje(1,1:50) = 1;
 ile_minucji=30;
for palec=1:1:5
    if palec == 1
       wspt_t = 0.8; %wspolczynnik rowny threshold
       krawedz_y =150; %odl. odciecia do usuwania minucji
       krawedz_x =130;
       wsp_medfilt=10
   end
   if palec == 1
       wspt_t = 0.9;
       krawedz_y =180;
       krawedz_x = 130;
       wsp_medfilt=3
   elseif palec == 2
       wspt_t = 0.8;
```

```
        krawedz_y = 180;
        krawedz_x = 200;
        wsp_medfilt=3
    elseif palec == 3
        wspt_t = 0.85;
        krawedz_y = 230;
        krawedz_x = 170;
        wsp_medfilt=3
    elseif palec == 4
        wspt_t = 0.85;
        krawedz_y = 215;
        krawedz_x = 175;
        wsp_medfilt=4
    elseif palec == 5
        wspt_t = 0.8;
        krawedz_y = 240;
        krawedz_x = 165;
        wsp_medfilt=4
    end


    for i=1:1:10
        mystring=['C:\Users\Danielu\Documents\Systemy
inteligentne\projekt5\skan\wycinki\',num2str(palec),'(',num2str(i),')','.bmp'];
         [obraz] = imread(mystring);

         idx=i+10*(palec-1)

         %binaryzacja
        binarny = im2bw(obraz, wspt_t);

        %odwrocenie kolorow
        odwrocony = imcomplement(binarny);

        se = strel('disk',1); % element strukturalny

        %otwarcie
        otwarcie1 = imopen(odwrocony, se);

        %zamknięcie
        zamkniecie1 = imclose(otwarcie1, se);

        %filtr medianowy
        m = medfilt2(zamkniecie1, [wsp_medfilt wsp_medfilt]);

        %szkieletyzacja
        szkielet = bwmorph(m, 'skel', inf);

        %obciecie galazek
        galazki = bwmorph(szkielet, 'spur', 4);

        pojedyncze = bwmorph(galazki, 'clean', 4);

        imshow(pojedyncze);
        hold on;
        %znajdowanie minucji:
        minucje = bwmorph(pojedyncze,'branchpoints');
```

```
        nr=1;

            for k=krawedz_y:1:length(minucje(:,1))-krawedz_y
                for s=krawedz_x:1:length(minucje(1,:))-krawedz_x
                    if ( (minucje(k,s) == 1)  )
                        x_minucje(nr,idx)=s;
                        y_minucje(nr,idx)=k;
                        nr=nr+1;
                    else
                    end
                end
            end



        plot(x_minucje(:,idx),y_minucje(:,idx), 'ro','MarkerFaceColor','w','MarkerSize',4);

        %obliczenie odległości od pierwszej wykrytej minucji
        for z=1:ile_minucji
            odleglosci(z,idx)=sqrt(  (((x_minucje(1,idx))  -   (x_minucje(z,idx))).^2 ) +
(((y_minucje(z,idx))  -   (y_minucje(1,idx))).^2 ) ) ;
        end
        % ilość znalezionych minucji na obrazie
        ilosc_minucji = length(y_minucje(:,idx));
        % średnia wartość współrzędniej Y z wszystkich minucji
        y_mean_minucji = mean(y_minucje(:,idx));
        % średnia wartość współrzędniej X z wszystkich minucji
        x_mean_minucji = mean(x_minucje(:,idx));
        % wariancja współrzędnej X Minucji
        x_minucji_var = var(x_minucje(:,idx));
        % odchylenie standardowe współrzędnej X minucji
        x_minucji_std = std(x_minucje(:,idx));
        % wariancja współrzędnej Y Minucji
        y_minucji_var = var(y_minucje(:,idx));
        % odchylenie standardowe współrzędnej Y minucji
        y_minucji_std = std(y_minucje(:,idx));
        % średnia odległość od pierwszej minucji
        sr_odl_od_1_M = mean(odleglosci(:,idx));
        % wariancja zbioru odległości od pierwszej minucji
        var_odl_od_1_M = var(odleglosci(:,idx));
        % odchylenie standardowe zbioru odległości od pierwszej minucji
        odch_odl_od_1_M = std(odleglosci(:,idx));

        %tworzenie macierzy uczącej
        dane(:,idx) = [ilosc_minucji; y_mean_minucji; x_mean_minucji;
x_minucji_var;y_minucji_var;y_minucji_std;x_minucje(1:ile_minucji,idx);y_minucje(1:ile_minu
cji,idx);sr_odl_od_1_M; var_odl_od_1_M;odch_odl_od_1_M];
        Mucz=dane;

    end
end

%NORMALIZACJA MACIERZY UCZĄCEJ:
 for ile_param = 1: length(Mucz(:,1))
            maximum(ile_param) = max(Mucz(ile_param, :));
            minimum(ile_param) = min(Mucz(ile_param, :));
                if maximum(ile_param) >= abs(minimum(ile_param))
```

```
            dzielnik(ile_param) = maximum(ile_param);
          else
            dzielnik(ile_param) = abs(minimum(ile_param));
          end
end

    for r=1:1:length(Mucz(:, 1))
        for j=1:1:length(Mucz(1, :))
        Mucz(r, j) = Mucz(r, j) / dzielnik(r);
        end
    end

probki_uczace= [1:7 11:17 21:27 31:37 41:47 ];
probki_testowe= [8:10 18:20 28:30 38:40 48:50];
M_ucz=Mucz(:,probki_uczace);
M_test=Mucz(:,probki_testowe);

liczba_probek_uczacych=7; %określa liczbe probek w kazdej serii

%DEFINIOWANIE MACIERZY CELU:
 M_cel = zeros(4, length(M_ucz(1, :)));
 kolejnosc= 1;
for j=1:1:length(M_ucz(1, :))
        M_cel(kolejnosc,j)=1;
        if mod(j,liczba_probek_uczacych)==0
           kolejnosc=kolejnosc+1;
        end
end

  %UCZENIE
  tic
  B=[0 1];
  PR=repmat(B,69,1); % 69 parametrów wejściowych
  mnet=newff(PR,[120 5],{'tansig' 'logsig'},'traingdx'); % 120 neuronów w warstwie ukrytej i 5
w warstwie wyjęciowej
  mnet=init(mnet);
  mnet.trainParam.show=10000;
  mnet.trainParam.epochs=6000;
  mnet.trainParam.goal=0;
  mnet=train(mnet,M_ucz,M_cel);
  toc
  Pegz=M_test;
  YY=sim(mnet,Pegz);
  yyt = YY';
```

Bibliografia

- *http://winntbg.bg.agh.edu.pl/skrypty/0001/0001.pdf*
- *https://pl.wikipedia.org/wiki/Linie_papilarne*
- *http://www.mathworks.com/help/matlab/*
- *http://sequoia.ict.pwr.wroc.pl/*
- *http://www.ftj.agh.edu.pl/~stegowski/rozne/neurony/art_kern_1.pdf*