Informatyka, studia dzienne, mgr II st. semestr I

Obliczenia inteligentne **2016/2017**

Prowadzący: Prowadzący dr inż. Arkadiusz Tomczyk Środa, 10:30

Data oddania: _____ Ocena: _____

Mateusz Grabowski 214903

Elementy twarzy - solution1 2.0

1. Wstęp

1.1. Cel

Przygotowana aplikacja ma za zadanie lokalizację wybranych elementów twarzy na czarno-białych zdjęciach, zawierających pojedyncze twarze. Poszukiwany zbiór liczy 20 elementów twarzy, ze spisem tych elementów można zapoznać się w dalszej części raportu.

1.2. Założenia

Podstawowym założeniem przy pracy nad tym zagadnieniem było zastosowanie sieci neuronowej do lokalizacji elementów twarzy.

1.3. Budowanie rozwiązania

Rozwiązanie zostało przygotowane w języku C++ z użyciem środowiska i bibliotek Qt, W związku z tym zbudowanie rozwiązania wymaga posiadania programów *qmake* i *make*. W tym celu należy wykonać następujące polecenia:

```
$ sudo apt-get install qt5-qmake
$ sudo apt-get install qtbase5-dev
$ export QT_SELECT=qt5
```

Aby zbudować rozwiązanie należy uruchomić skrypt *build.sh* zamieszczony w katalogu rozwiązania, a przed uruchomieniem środowiska testującego należy wykonać skrypt *deploy.sh*.

2. Rozwiązanie wstępne

2.1. Opis

Rozwiązanie zostało oparte o wielowarstwową jednokierunkową sieć neuronową. Sieć neuronowa składa się z neuronów typu Adaline, z liniową i sigmoidalną funkcją aktywacji.

Sieć ma strukturę trójwarstwową:
— Warstwa wejściowa zawiera 6114 neuronów, z liniową funkcją aktywacji
— Warstwa ukryta zawiera 150 neuronów, z sigmoidalną funkcją aktywacji
— Warstwa wyjściowa zawiera 40 neuronów, z liniową funkcją aktywacji
Sieć ta była nauczana metodą wstecznej propagacji, w oparciu o dane dostarczone razem z bazą danych.

1) Wybierz niewielką wartość kroku nauki η (np. $\eta = 0.7$), zaś początkowe wartości wszystkich wag sieci wybierz jako małe liczby losowe (np. z przedziału [-1 , 1]) .

2) Wybierz losowo wzorzec uczący ($[\, v^{(0)}_{1\mu_0} \dots v^{(0)}_{m_0\,\mu_0}\,], [\, z_{1\mu_0} \dots z_{m_n\mu_0}\,]) \in \Omega$; $\mu_0 \in \{1, \dots, N\}$ ze zbioru treningowego Ω i przepuść sygnały wejściowe $v^{(0)}_{1\mu_0}, \dots, v^{(0)}_{m_0\,\mu_0}$ przez sieć w przód wyznaczając i zapamiętując wyjścia $v^{(k)}_{i\mu_0}$ i sumy ważone $s^{(k)}_{i\mu_0}$ dla wszystkich neuronów sieci.

3) Oblicz sygnały zwrotne $\delta^{(n)}_{i\mu_0}$ wszystkich neuronów warstwy wyjściowej sieci korzystając ze wzoru:

$$\delta^{(n)}_{i\mu_0} = f'(s^{(n)}_{i\mu_0}) \cdot (\, z_{i\mu_0} - v^{(n)}_{i\mu_0}) \,; \quad i = 1, \dots, m_n$$

4) Oblicz sygnały zwrotne $\delta^{(k)}_{i\mu_0}$ wszystkich neuronów warstw poprzednich sieci propagując te sygnały kolejno wstecz sieci poczynając od warstwy $n-1$ aż do warstwy1 za pomocą wzoru:

$$\delta^{(k)}_{i\mu_0} = f'(s^{(k)}_{i\mu_0}) \cdot \sum_{j=1}^{m_{k+1}} w^{(k+1)}_{ji}\, \delta^{(k+1)}_{j\mu_0} \,; \quad i = 1, \dots, m_k \,; \quad k = n-1, \dots, 1$$

5) Korzystając z wyznaczonych i zapamiętanych w punktach **2) - 4)** wielkości wyjść $v^{(k)}_{i\mu_0}$ i sygnałów zwrotnych $\delta^{(k)}_{i\mu_0}$ neuronów sieci dokonaj zmiany każdej z wag uczonej sieci według wzoru:

$$w^{(k)}_{pq} = w^{(k)}_{pq} + \eta\, \delta^{(k)}_{p\mu_0}\, v^{(k-1)}_{q\,\mu_0} \,; \quad p = 1, \dots, m_k \,; \quad q = 1, \dots, m_{k-1} \,; \quad k = 1, \dots, n$$

6) Jeśli nie wyczerpano wszystkich wzorców uczących ze zbioru Ω to wybierz losowo kolejny wzorzec, nie podawany jeszcze na wejście sieci, i przejdź do punktu **2)**, w przeciwnym razie idź do punktu **7)**.

7) Czy sieć odtwarza z założoną dokładnością każdy ze wzorców treningowych? Jeśli tak to koniec. W przeciwnym razie rozpocznij kolejną *epokę* nauczania sieci przechodząc do punktu **2)**.

Rysunek 1. Algorytm wstecznej propagacji błędu.

2.2. Wstępne przetwarzanie obrazów

Ze względu na optymalizację czasu nauki sieci niezbędne było zmniejszenie liczby wejść sieci neuronowej. W tym celu wartości wejść zostały wybrane w następujący sposób:

1. Ponieważ osoby na obrazach zawsze występują centralnie na obrazie, podjąłem decyzję o horyzontalnym podziale obrazu na 4 ćwiartki i odrzuceniu dwóch skrajnych, pozostawiając jedynie środek. Pozwoliło to zmniejszyć liczbę pikseli o połowę.
2. Następnie następnie z otrzymanego fragmentu obrazu zostały wybrane piksele, które spełniały warunek: co trzeci piksel w danym wierszu, w co trzecim wierszu.

Takie podejście pozwoliło zmniejszyć rozmiar warstwy wejściowej około 16x.

Dodatkowo wartości wyselekcjonowanych pikseli zostały przeskalowane do wartości z zakresu [0, 1].

2.3. Wyniki

Listing 1. Ocena rozwiązania podstawowego.

```
Square mean error:
        0. RIGHT_EYE_PUPIL_ERROR:                0.282275
        1. LEFT_EYE_PUPIL_ERROR:                 0.296373
        2. RIGHT_MOUTH_ERROR:                    0.341085
        3. LEFT_MOUTH_ERROR:                     0.343877
        4. RIGHT_OUTER_EYEBROW_ERROR:            0.306529
        5. RIGHT_INNER_EYEBROW_ERROR:            0.284366
        6. LEFT_INNER_EYEBROW_ERROR:             0.295697
        7. LEFT_OUTER_EYEBROW_ERROR:             0.338969
        8. RIGHT_TEMPLE_ERROR:                   0.290888
        9. RIGHT_OUTER_EYE_ERROR:                0.289217
        10. RIGHT_INNER_EYE_ERROR:               0.285033
        11. LEFT_INNER_EYE_ERROR:                0.292133
        12. LEFT_OUTER_EYE_ERROR:                0.300942
        13. LEFT_TEMPLE_ERROR:                   0.317229
        14. TIP_OF_NOSE_ERROR:                   0.313311
        15. RIGHT_NOSTRIL_ERROR:                 0.314611
        16. LEFT_NOSTRIL_ERROR:                  0.322415
        17. CENTRE_POINT_OF_UPPER_LIP_ERROR:     0.346198
        18. CENTRE_POINT_OF_LOWER_LIP_ERROR:     0.339738
        19. TIP_OF_CHIN_ERROR:                   0.37376

Correct location of all elements: 1.97368 %

        RIGHT_EYE_PUPIL_SUCCESS:                 24.1776 %
        LEFT_EYE_PUPIL_SUCCESS:                  23.0263 %
        RIGHT_MOUTH_SUCCESS:                     14.1447 %
        LEFT_MOUTH_SUCCESS:                      17.9276 %
        RIGHT_OUTER_EYEBROW_SUCCESS:             21.875 %
        RIGHT_INNER_EYEBROW_SUCCESS:             24.6711 %
        LEFT_INNER_EYEBROW_SUCCESS:              21.875 %
        LEFT_OUTER_EYEBROW_SUCCESS:              15.4605 %
        RIGHT_TEMPLE_SUCCESS:                    24.5066 %
        RIGHT_OUTER_EYE_SUCCESS:                 24.8355 %
        RIGHT_INNER_EYE_SUCCESS:                 24.3421 %
        LEFT_INNER_EYE_SUCCESS:                  23.8487 %
        LEFT_OUTER_EYE_SUCCESS:                  23.0263 %
        LEFT_TEMPLE_SUCCESS:                     20.7237 %
        TIP_OF_NOSE_SUCCESS:                     17.4342 %
        RIGHT_NOSTRIL_SUCCESS:                   16.4474 %
        LEFT_NOSTRIL_SUCCESS:                    15.9539 %
        CENTRE_POINT_OF_UPPER_LIP_SUCCESS:       14.6382 %
        CENTRE_POINT_OF_LOWER_LIP_SUCCESS:       13.4868 %
        TIP_OF_CHIN_SUCCESS:                     13.9803 %
```

3. Ewolucja rozwiązania

W tym rozdziale zostały zamieszczone kolejne (w kolejności chronologicznej) próby modyfikacji rozwiązania, które miały na celu poprawę wyników otrzymanych przez rozwiązanie wstępne.

3.1. Analiza parametrów nauki sieci neuronowej

3.1.1. Założenia

Podczas pracy sieciami neuronowymi, nierozłącznym elementem jest nauka takiej sieci. W związku z czym parametry nauki takiej sieci mają istotny wpływ na otrzymywane wyniki. Tym samym znalezienie optymalnych parametrów powinno przełożyć się na lepsze wyniki lokalizacji elementów.

3.1.2. Wnioski

W ramach analizy wyznaczyłem zbiór parametrów mających istotny wpływ na proces nauki i wyniki sieci neuronowej. Parametry to:
— Krok nauki - około 0.0001 - 0.00001.
— Krok momentum - 0.1.
— Liczba neuronów w warstwie ukrytej - 20 - 80.
— Parametr *Beta* w funkcji sigmoidalnej neuronów w warstwie ukrytej - 0.01 - 5.

Odpowiednie dobranie wymienionych parametrów jest niezbędne, aby sieć była w stanie nauczyć się badanego wzorca. A także wpływa na wyniki lokalizacji elementów twarzy na zdjęciach testowych.

W przypadku gdy parametr Beta funkcji sigmoidalnej jest bliski jedności obserwujemy, że wyjścia warstwy ukrytej mają charakter binarny. W takiej sytuacji liczba neuronów w warstwie ukrytej może być mniejsza. W przypadku gdy parametr Beta ma wartości około 0.01, wyjścia warstwy ukrytej mają wartości z przedziału [0, 1], aby warstwa wyjściowa poprawnie je zinterpretowała niezbędna jest większa liczba neuronów w warstwie ukrytej. Różnica pomiędzy liczbą neuronów dla tych dwóch podejść może sięgać nawet około 3 razy. Dodatkowo mały parametr Beta przekłada się na mniej stabilny proces uczenia się sieci.

Analiza ta musi zostać przeprowadzona po każdej modyfikacji rozwiązania, albowiem raz dobrane parametry mogą być nieodpowiednie (otrzymujemy słabe wyniki) lub złe (sieć nie może nauczyć się wzorca) dla danych, które zostały zmodyfikowane w jakiś sposób. W związku z czym ten krok musiał być wykonywany przy każdej kolejnej modyfikacji rozwiązania.

3.2. Mieszanie elementów zbioru uczącego dla każdej epoki

3.2.1. Założenia

Uczenie sieci neuronowej stałym (ta sama kolejność) zbiorem zdjęć uczącym może doprowadzić do wyuczenia się sieci pewnego wzorca, przez co wyniki otrzymywane na zbiorze testowym mogą być gorsze, ponieważ nie zawierają wyuczonego wzorca. Mieszanie zbioru uczącego powinno uelastycznić uczącą sieć, co powinno prowadzić do polepszenia wyników.

3.2.2. Wnioski

Dla rozwiązania wstępnego wprowadzenie mieszania zbioru uczącego sprawiła, że rozwiązanie nie jest w stanie prawidłowo wykryć elementów twarzy na żadnym zdjęciu. Otrzymany wynik był dość zaskakujący i nie napawał optymizmem.

Po dłuższej analizie doszedłem do wniosku, że mieszanie elementów zbioru uczącego w połączeniu z kolejnymi modyfikacjami poprawiło średnią lokalizację elementów twarzy o około 2-3%. Proces ten ustabilizował także proces nauki sieci neuronowej, tym samym zwiększyła się szansa, że sieć nauczyć wzorca.

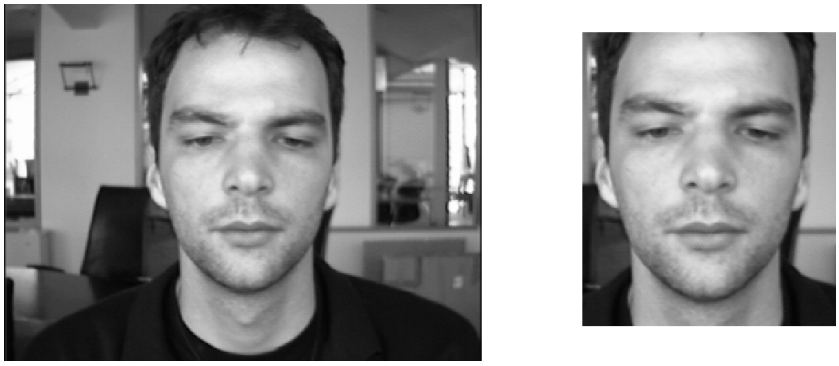
3.3. Ograniczenie przetwarzania do wycinka zdjęcia zawierającego twarz

3.3.1. Założenia

Przedmiotem pracy jest wykrywanie elementów twarzy, w związku z czym obszary zdjęcia nie zawierające twarzy są zbędne. Ograniczenie badanego obszaru do wycinka zawierającego twarz powinno poprawić lokalizację elementów, a także skrócić czas nauki i wykonywania programu (mniejsza ilość pikseli do przetworzenia).

W celu wykrycia położenia twarzy zakładam użycie biblioteki OpenCV i oczekuje otrzymanie położenia i rozmiaru prostokąta zawierającego twarz dla każdego zdjęcia z zbioru uczącego i testowego.

3.3.2. Wyniki



Rysunek 2. Lewe zdjęcie - oryginał, Prawe zdjęcie - wycinek zdjęcia zawierający twarz.

3.3.3. Wnioski

Implementacja tego kroku w połączeniu z następnym 3.4 znacznie poprawiło możliwości sieci w zakresie lokalizacji elementów twarzy.

Nie mniej jednak niezbędne było wykonanie dodatkowego kroku, który polegał na przesunięciu położeń elementów wczytanych z plików adnotacji. Spowodowane to było faktem, że wycięte fragmenty nie zawierają informacji o położeniu twarzy względem całego zdjęcia. Po uwzględnieniu tego faktu sieć neuronowa była w stanie wyuczyć się wzorca.

3.4. Skalowanie wycinka zdjęcia zawierającego twarz

3.4.1. Założenia

Sieć neuronowa ma określoną liczbę wejść, w związku z czym niezbędne było przeskalowanie otrzymanych wycinków twarzy (z kroku 3.3) do określonego rozmiaru. Skalowanie obrazu do rozmiaru mniejszego wiąże się z stratą pewnej ilości informacji z zdjęcia. W związku z czym należy zbadać zależność między rozmiarem badanego okna, a wynikami sieci.

3.4.2. Wyniki

Rysunek 3. Skalowania odpowiednio: oryginał, 100x100, 64x64, 32x32 pikseli.

Listing 2. Średnia ocena dla poszczególnych skal.

```
Srednia ocena dla skali 100x100: 17% +/- 2%
Srednia ocena dla skali 64x64: 18% +/- 3%
Srednia ocena dla skali 32x32: 19% +/- 5%
```

3.4.3. Wnioski

Podczas eksperymentów można zauważyć pewien trend. Mianowicie obrazy w mniejszej skali mają większą rozbieżność od średniej, ale otrzymujemy średnio lepsze wyniki. Natomiast większe obrazy zachowują większą stabilność co do wyników. Należy jednak zwrócić uwagę, że wyniki podane na listingu 2 są obarczone dużym błędem wynikającym z dość małego zbioru przeprowadzonych testów. Było to spowodowane długim czasem uczenia sieci.

Ostatecznie wybrany rozmiar okna to 32x32 piksele. Poza dobrymi wynikami tego okna, istotne było zredukowanie czasu uczenia sieci neuronowej. Spowodowane to było redukcją liczby wejść sieci, ze względu na mniejszą ilość pikseli.

3.5. Zastosowanie algorytmu wykrywania krawędzi

3.5.1. Założenia

Kolejnym krokiem redukcji zbędnych informacji z fragmentu obrazy jest wykrycie krawędzi. Jest to podyktowane faktem, że szukane elementy twarzy znajdują się na krawędziach lub ich okolicach. Efekt ten planuje uzyskać stosując operator Sobela.

3.5.2. Wyniki

Rysunek 4. Algorytmu wykrywania krawędzi odpowiednio dla: 100x100, 64x64, 32x32 pikseli.

Listing 3. Średnia ocena dla poszczególnych skal z zastosowaniem algorytmu wykrywania krawędzi.

```
Srednia ocena dla skali 100x100: 21% +/- 3% (max 24%)
Srednia ocena dla skali 64x64: 22% (max 25%)
Srednia ocena dla skali 32x32: 21% +/- 3% (max 28%)
```

3.5.3. Wnioski

Implementacja algorytmu wykrywania krawędzi przyczyniła się do poprawy lokalizacji elementów twarzy. Nie mniej jednak spodziewałem się większej poprawy.

Niewielka poprawa może wynikać z faktu dojścia do granicy polepszania lokalizacji metodą redukcji zbędnych informacji z zdjęcia. Inną propozycją mogą być szumy, pozostawione po zastosowaniu algorytmu wykrywania krawędzi.

3.6. Modyfikacja kontrastu obrazu

3.6.1. Założenia

Zwiększenie kontrastu obrazu miało na celu wyostrzenie krawędzi wykrytych w poprzednim kroku i usunięciu szumu. W tym celu zakładałem modyfikację kontrastu obrazu w pewnym zakresie, skupiając się na jego zwiększeniu.

3.6.2. Wyniki

Rysunek 5. Modyfikacja kontrastu odpowiednio dla: 100x100, 64x64, 32x32 pikseli.

Listing 4. Średnia ocena dla poszczególnych skal z zastosowaniem algorytmu wykrywania krawędzi i zwiększeniu kontrastu.

```
Srednia ocena dla skali 100x100: 17%
Srednia ocena dla skali 64x64: 18% +/- 3%
Srednia ocena dla skali 32x32: 0%
```

3.6.3. Wnioski

Modyfikacja kontrastu przyczyniła się do pogorszenia zdolności lokalizacji elementów twarzy. Dla skali 32x32 pikseli, sieć nie była w stanie wyuczyć się wzorca. Otrzymany rezultat może poprzeć postawioną w podrozdziale 3.5.3 tezę, o osiągnięciu granicy poprawy wyników z użyciem podejścia opierającego się na redukcji zbędnych informacji z obrazu.

4. Wyniki

Prezentowany wynik został uzyskany dla sieci o 1024 wejściach, 60 neuronów w warstwie ukrytej oraz 40 wyjściach. Parametr *Beta* równy 5, oraz rozmiar okna 32x32 pikseli.

Listing 5. Ostateczny rezultat lokalizacji elementów twarzy.

```
Square mean error:
        0. RIGHT_EYE_PUPIL_ERROR:           0.136434
        1. LEFT_EYE_PUPIL_ERROR:            0.223968
        2. RIGHT_MOUTH_ERROR:               0.223454
        3. LEFT_MOUTH_ERROR:                0.277977
        4. RIGHT_OUTER_EYEBROW_ERROR:       0.111165
        5. RIGHT_INNER_EYEBROW_ERROR:       0.145614
        6. LEFT_INNER_EYEBROW_ERROR:        0.190227
        7. LEFT_OUTER_EYEBROW_ERROR:        0.256331
        8. RIGHT_TEMPLE_ERROR:              0.128348
        9. RIGHT_OUTER_EYE_ERROR:           0.128182
        10. RIGHT_INNER_EYE_ERROR:          0.152423
        11. LEFT_INNER_EYE_ERROR:           0.206711
        12. LEFT_OUTER_EYE_ERROR:           0.244411
        13. LEFT_TEMPLE_ERROR:              0.268234
        14. TIP_OF_NOSE_ERROR:              0.213813
        15. RIGHT_NOSTRIL_ERROR:            0.206371
        16. LEFT_NOSTRIL_ERROR:             0.220419
        17. CENTRE_POINT_OF_UPPER_LIP_ERROR:    0.233783
        18. CENTRE_POINT_OF_LOWER_LIP_ERROR:    0.261305
        19. TIP_OF_CHIN_ERROR:              0.31403

Correct location of all elements: 24.0132 %

        RIGHT_EYE_PUPIL_SUCCESS:            68.2566 %
        LEFT_EYE_PUPIL_SUCCESS:             46.0526 %
        RIGHT_MOUTH_SUCCESS:                49.5066 %
        LEFT_MOUTH_SUCCESS:                 36.3487 %
        RIGHT_OUTER_EYEBROW_SUCCESS:        83.2237 %
        RIGHT_INNER_EYEBROW_SUCCESS:        65.1316 %
        LEFT_INNER_EYEBROW_SUCCESS:         52.7961 %
        LEFT_OUTER_EYEBROW_SUCCESS:         40.625 %
        RIGHT_TEMPLE_SUCCESS:               72.5329 %
        RIGHT_OUTER_EYE_SUCCESS:            71.3816 %
        RIGHT_INNER_EYE_SUCCESS:            62.6645 %
        LEFT_INNER_EYE_SUCCESS:             49.1776 %
        LEFT_OUTER_EYE_SUCCESS:             42.9276 %
        LEFT_TEMPLE_SUCCESS:                40.1316 %
        TIP_OF_NOSE_SUCCESS:                44.7368 %
        RIGHT_NOSTRIL_SUCCESS:              49.6711 %
        LEFT_NOSTRIL_SUCCESS:               45.7237 %
        CENTRE_POINT_OF_UPPER_LIP_SUCCESS:      44.7368 %
        CENTRE_POINT_OF_LOWER_LIP_SUCCESS:      37.3355 %
        TIP_OF_CHIN_SUCCESS:                30.2632 %
```

5. Wnioski

— Stworzone rozwiązanie prawidłowo zlokalizowało wszystkie elementy twarzy na średnio 22% procentach zdjęć.
— Biorąc pod uwagę liczbę wszystkich elementów (20) uważam to za dość dobry wynik.
— Otrzymany wynik jest głównie ograniczony przez zdolność lokalizacji elementu o najmniejszym procencie wykrycia. Element ten można znaleźć na listingu 5.
— Pomimo, że otrzymany rezultat lokalizacji wszystkich elementów na twarzy jest niski, to statystyki lokalizacji pojedynczych elementów są bardziej zachęcające.
— Najgorzej lokalizowany jest broda. Może to wynikać z faktu małej ilości informacji (krawędzi) w tym obszarze.
— Problematyczne są zdjęcia na których twarz ma mały rozmiar, w takich przypadkach otrzymany wynik zwykle jest w większej skali, przez co część elementów oddala się od prawdziwych miejsc. Może się to wiązać z brakiem/małą ilością podobnych zdjęć w zbiorze testowym.
— Na listingu możemy dostrzec jeszcze jedną prawidłowość, mianowicie lokalizacja elementów twarzy znajdujących się po prawej stronie jest prawie 2 razy lepsza niż tych z lewej, Taki wynik może sugerować, że w miarę oddalania się od prawej strony twarzy, pozostałe elementy są nieproporcjonalnie oddalone.