Sieć Kohonena - Metody inteligencji obliczeniowej w analizie danych

Piotr Bielecki,nr albumu 313320

1 Wstęp

1.1 Cel

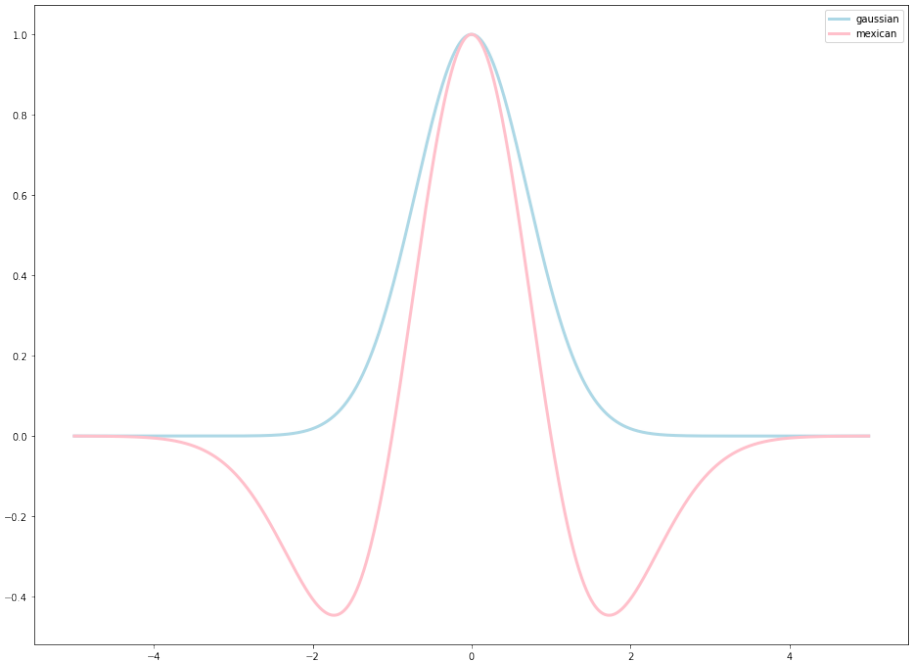
Raport poświęcony jest opracowaniu wyników pracy z przedmiotu "Metody inteligencji obliczeniowej w analizie danych w temacie samoorganizujących się map znanych jako sieci Kohonena.

2 KOH1 - Podstawowa sieć Kohonena

2.1 cel ćwiczenia

celem tej pracy domowej, było stworzenie sieci Kohonena złożonej z neuronów w prostokątnej siatce, której wymiary są parametrami programu. Należało zaimplementować dwie różne funkcje sąsiedztwa:

- funkcję gaussowską
- minus drugą pochodną gaussowskiej (kapelusz meksykański)



Rysunek 1: funkcje sąsiedztwa

W obu implementacjach należało dodać możliwość zmiany szerokości sąsiedztwa z użyciem parametru i sprawdzić dla kilku wartości z przedziału [0.1,1].

Należało dodatkowo przetestować działanie sieci na dostarczonych prostych zbiorach danych:

- danych 2d skupionych w wierzchołkach sześciokąta
- danych 3d skupionych w wierzchołkach sześcianu

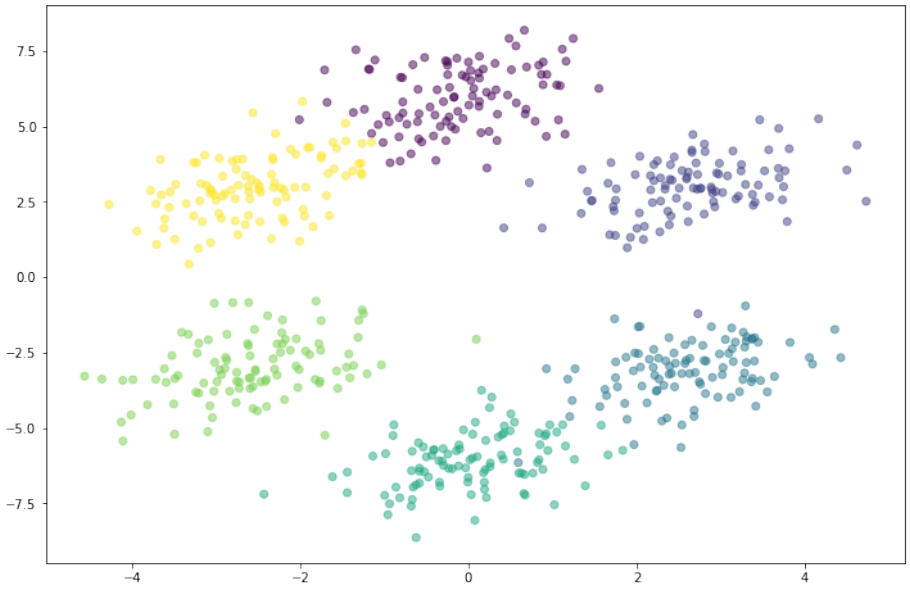
Jako funkcji wygaszającą uczenie wraz z kolejnymi iteracjami użyto funkcji:

$$\alpha(t) = \frac{et}{\lambda}$$

2.2 Co to jest sieć Kohonena?

Sieć Kohonena to rodzaj sieci neuronowej, której celem jest grupowanie danych wejściowych w przestrzeni wielowymiarowej. Sieć Kohonena składa się z neuronów ułożonych w siatce, połączonych ze sobą w nieskierowany sposób. Każdy neuron ma swoją wagę, która określa jego położenie w przestrzeni rozpatrywanego problemu. W czasie treningu sieć modyfikuje wagi, w odpowiedzi na dane wejściowe, w celu pogrupowania w klastry.

2.3 opracowanie, wnioski, wyniki

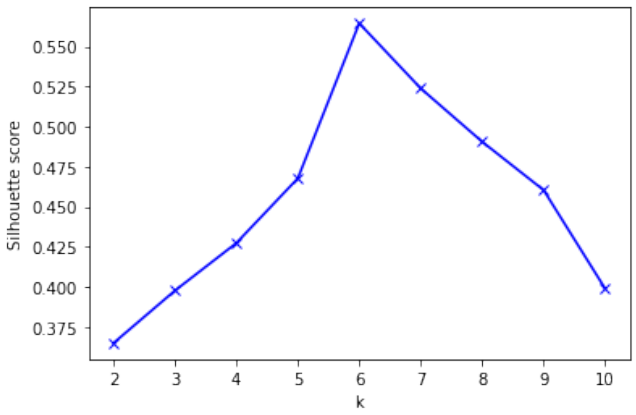


Rysunek 2: dane hexagon

2.3.1 hexagon - funkcja gaussowska

Czy jesteśmy w stanie znaleźć liczbę klastrów?

Zaprojektowany algorytm, po dopasowaniu do danych powyżej został zbadany przy pomocy miary silhouette w celu zbadania, czy poprawnie jesteśmy w stanie wyznaczyć liczbę klastrów. Maksimum funkcji wyraźnie wskazuje na 6.



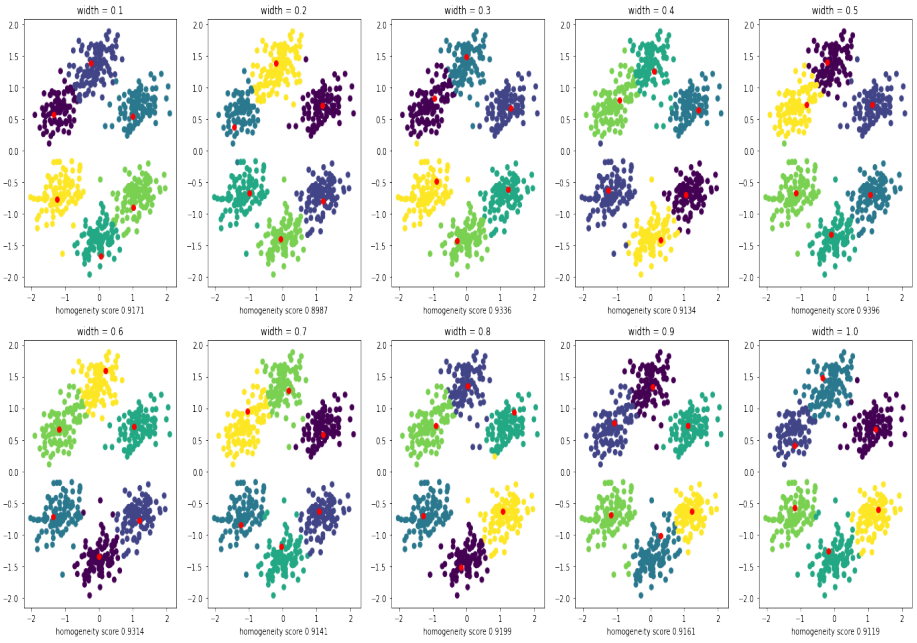
Rysunek 3:

Jak parametr width wpływa na wyniki uczenia?

width	min dist between clusters	mean dist in clust	mean dist to clust center	silhouette
0.1	0.058286	0.497787	0.348049	0.576424
0.2	0.002472	0.505322	0.352191	0.574786
0.3	0.054569	0.500681	0.349012	0.576851
0.4	0.061641	0.499491	0.348266	0.574145
0.5	0.050564	0.496932	0.347479	0.572681
0.6	0.051858	0.496385	0.354878	0.569683
0.7	0.039725	0.496411	0.347397	0.574697
0.8	0.061560	0.496018	0.350769	0.572228
0.9	0.013313	0.496499	0.348670	0.575371
1.0	0.034800	0.500174	0.348746	0.569578

Tabela 1: porównanie ze względu na parametr width

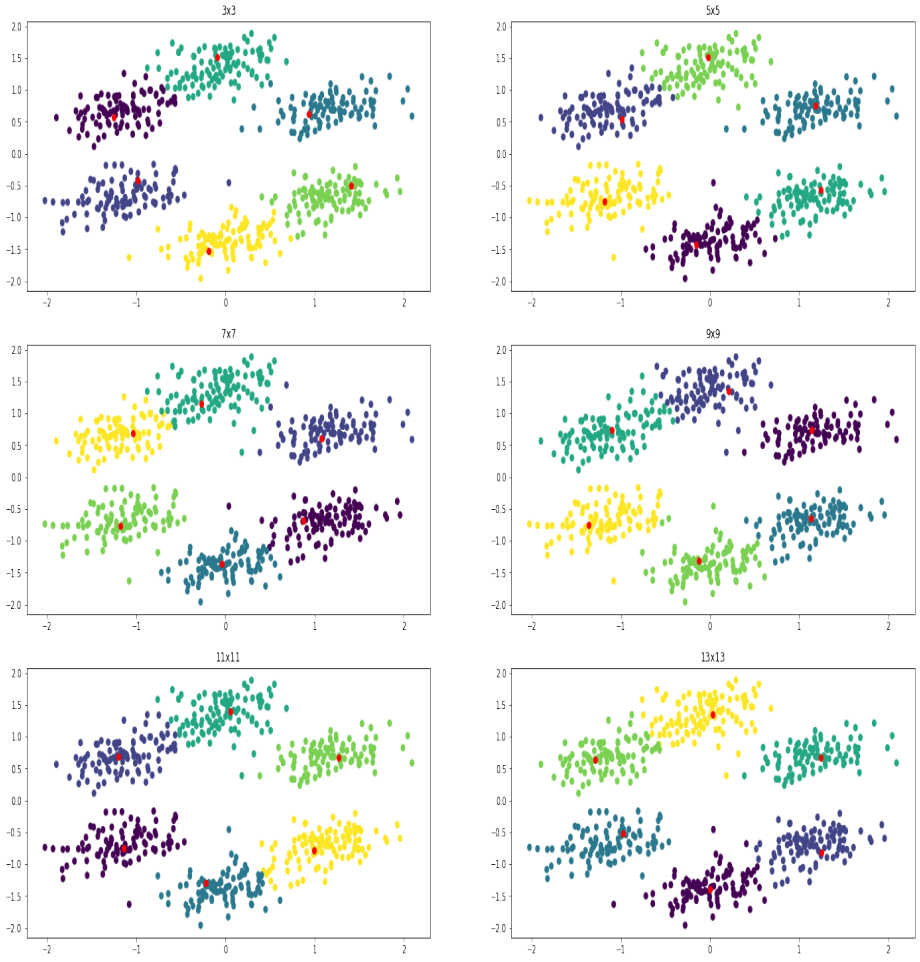
wartość parametru nie gra w tym problemie dużej roli, jedyną znaczącą wariancję w wynikach obserwujemy w kolumnie minimalnych odległości między klastrami.



Rysunek 4: klasteryzacja po uczeniu, na czerwono - wagi neuronów

Taka sama sytuacja - bez różnic w jakości dopasowania.

A jak z zadaniem radzą sobie różne rozmiary map?
Ponownie, nie widać szczególnych różnic jakościowych w klastrach dobranych przez sieci z różnymi architekturami.

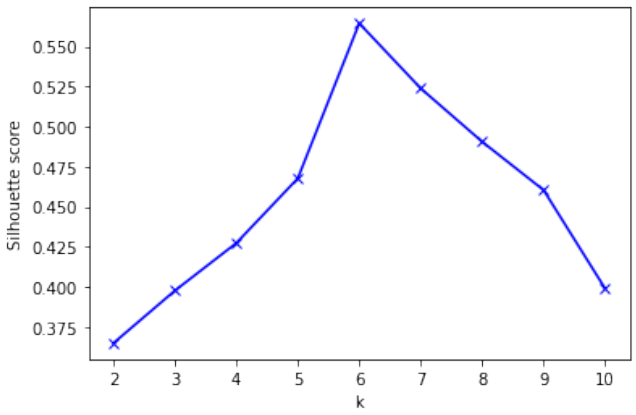


Rysunek 5: porównanie w zależności od rozmiaru sieci

2.3.2 hexagon - kapelusz meksykański

Czy jesteśmy w stanie znaleźć liczbę klastrów?

Ponownie, poprawnie wskazujemy odpowiednią liczbę klastrów.



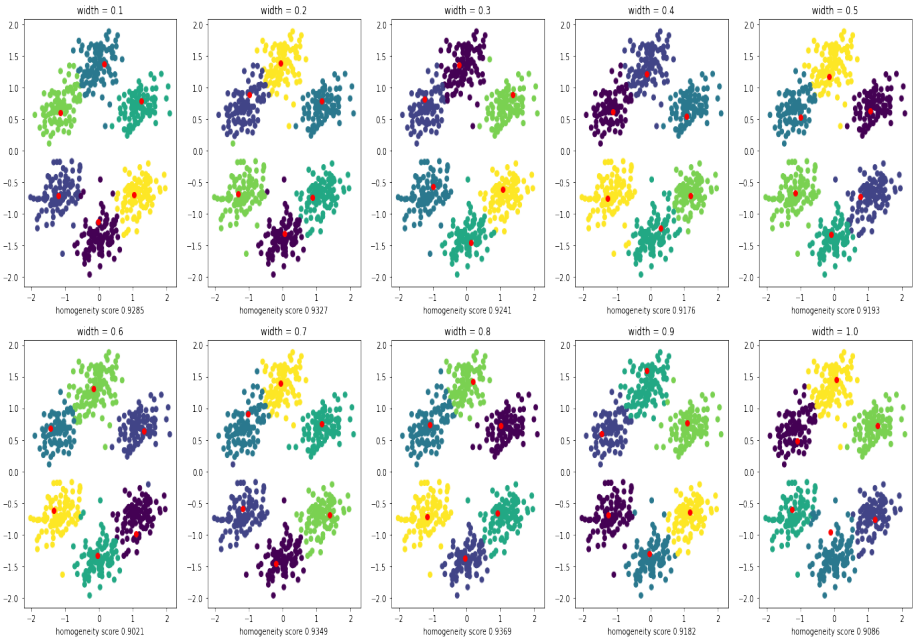
Rysunek 6:

Jak parametr width wpływa na wyniki uczenia?

width	min dist between clusters	mean dist in clust	mean dist to clust center	silhouette
0.1	0.026658	0.495682	0.349422	0.570291
0.2	0.068736	0.494504	0.347889	0.568470
0.3	0.039647	0.496220	0.347442	0.572492
0.4	0.021925	0.495326	0.347903	0.567826
0.5	0.023613	0.498317	0.348810	0.573874
0.6	0.031636	0.502386	0.347805	0.573766
0.7	0.031636	0.494899	0.364282	0.570288
0.8	0.060559	0.506329	0.351851	0.568759
0.9	0.047689	0.501165	0.346793	0.568146
1.0	0.055282	0.495951	0.350044	0.570307

Tabela 2:

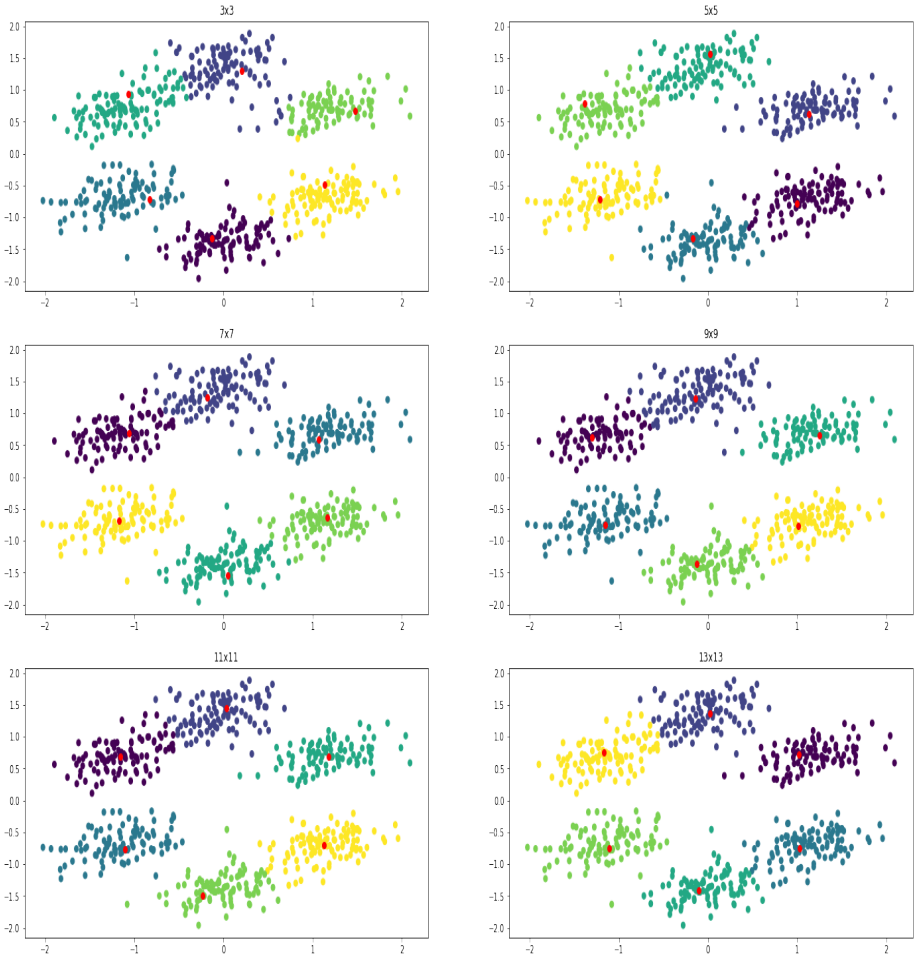
Identyczna sytuacja jak poprzednio, jedyną zmienność obserwujemy w miarze minimalnej odległości między klastrami.



Rysunek 7: klasteryzacja po uczeniu, na czerwono - wagi neuronów

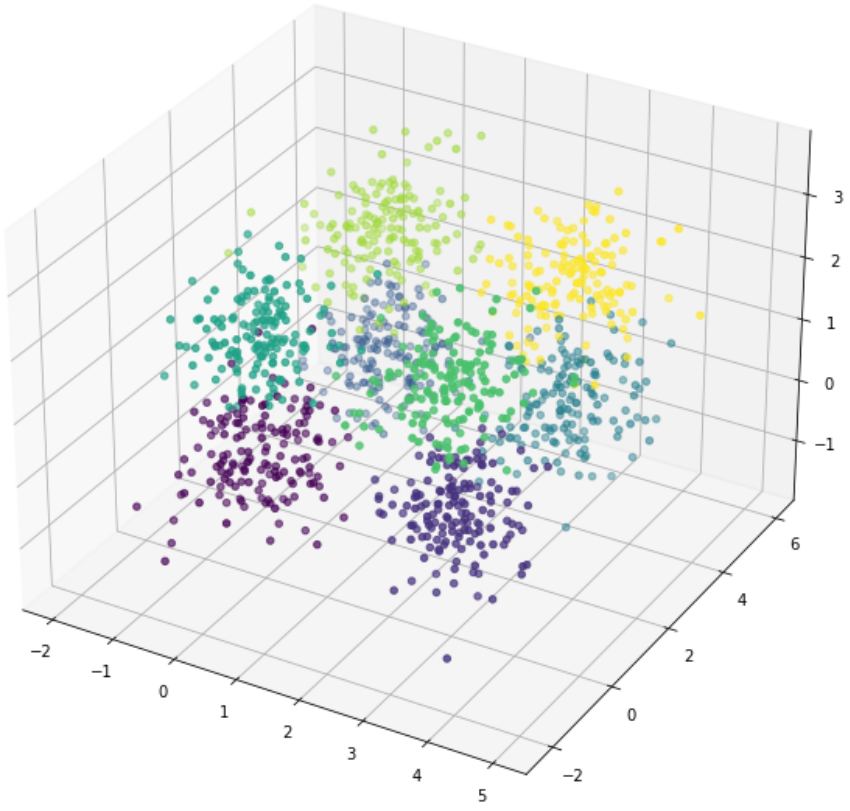
Prawie identycznie wyglądają klastry przyporządkowane przez sieć dla kolejnych wartości parametru.

A jak z zadaniem radzą sobie różne rozmiary map?
Ponownie, nie widać szczególnych różnic jakościowych w klastrach dobranych przez sieci z różnymi architekturami.



Rysunek 8: porównanie w zależności od rozmiaru sieci

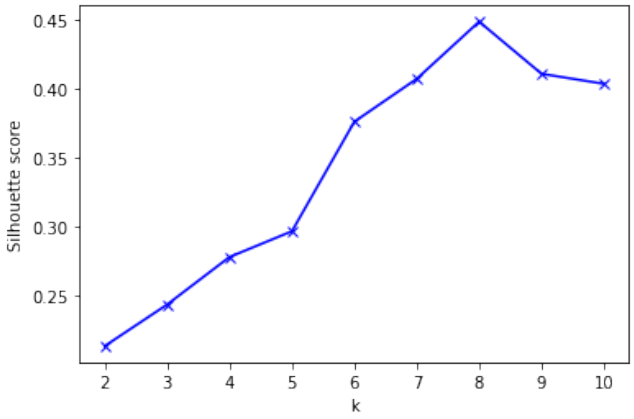
2.3.3 cube - funkcja gaussowska



Rysunek 9: dane cube

Czy jesteśmy w stanie znaleźć liczbę klastrów?

Zaprojektowany algorytm, po dopasowaniu do danych powyżej, jak w przypadku danych hexagon, został zbadany miarą silhouette. Ponownie, poprawnie identyfikujemy liczbę klastrów.



Rysunek 10:

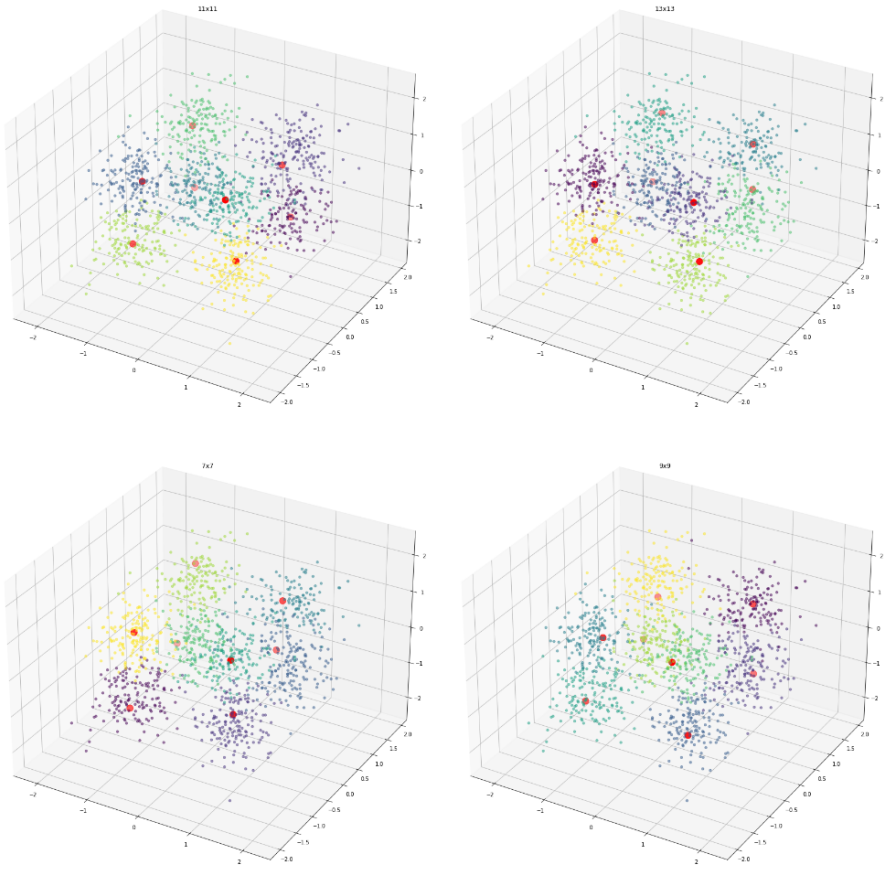
Jak parametr width wpływa na wyniki uczenia?

width	min dist between clusters	mean dist in clust	mean dist to clust center	silhouette
0.1	0.052589	0.856547	0.598857	0.476550
0.2	0.093270	0.854540	0.601544	0.470380
0.3	0.082029	0.848125	0.598220	0.472279
0.4	0.093270	0.852686	0.600202	0.459240
0.5	0.093270	0.858319	0.606861	0.480046
0.6	0.081282	0.849424	0.597989	0.474584
0.7	0.068117	0.846915	0.597806	0.469646
0.8	0.080178	0.852540	0.602680	0.444823
0.9	0.096924	0.850118	0.601093	0.472740
1.0	0.035233	0.862558	0.597096	0.447812

Tabela 3:

Identyczna sytuacja jak w poprzednich przypadkach, nieznaczne różnice w wartościach miar, dopasowane klastry nie różnią się wizualnie.

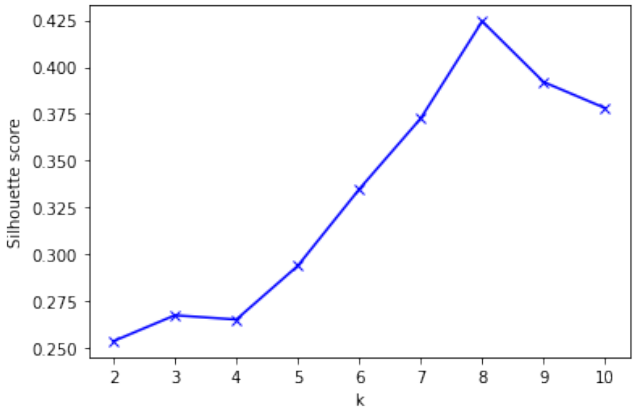
A jak z zadaniem radzą sobie różne rozmiary map?
Ponownie, nie widać szczególnych różnic jakościowych w klastrach dobranych przez sieci z różnymi architekturami.



Rysunek 11: porównanie w zależności od rozmiaru sieci, od lewej z góry 11x11, 13x13, 7x7, 9x9

2.3.4 hexagon - kapelusz meksykański

Czy jesteśmy w stanie znaleźć liczbę klastrów?
Ponownie, poprawnie wskazujemy odpowiednią liczbę klastrów.



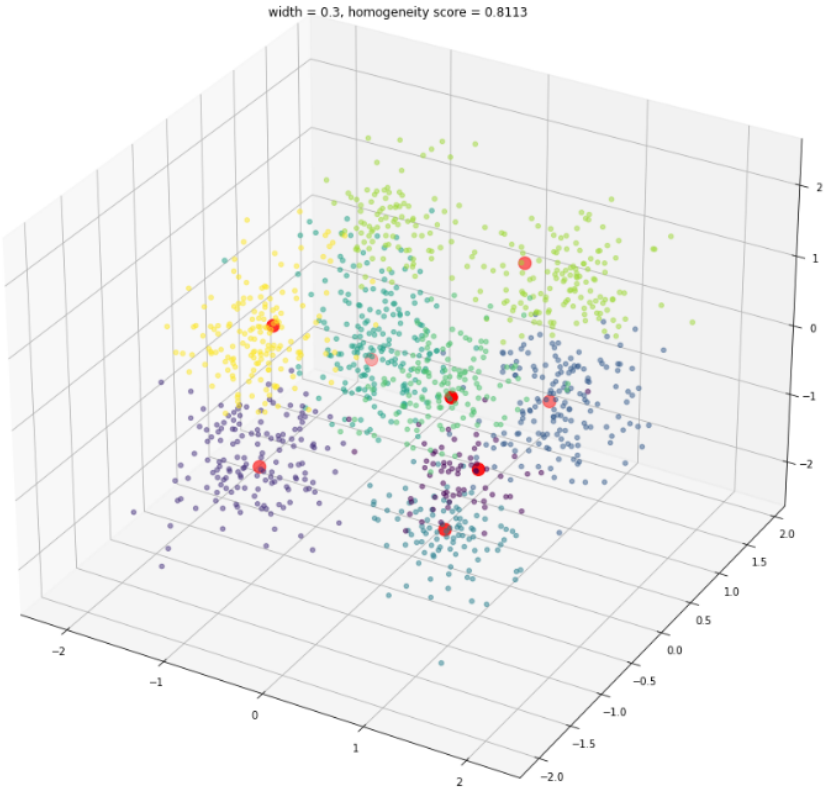
Rysunek 12:

Jak parametr width wpływa na wyniki uczenia?

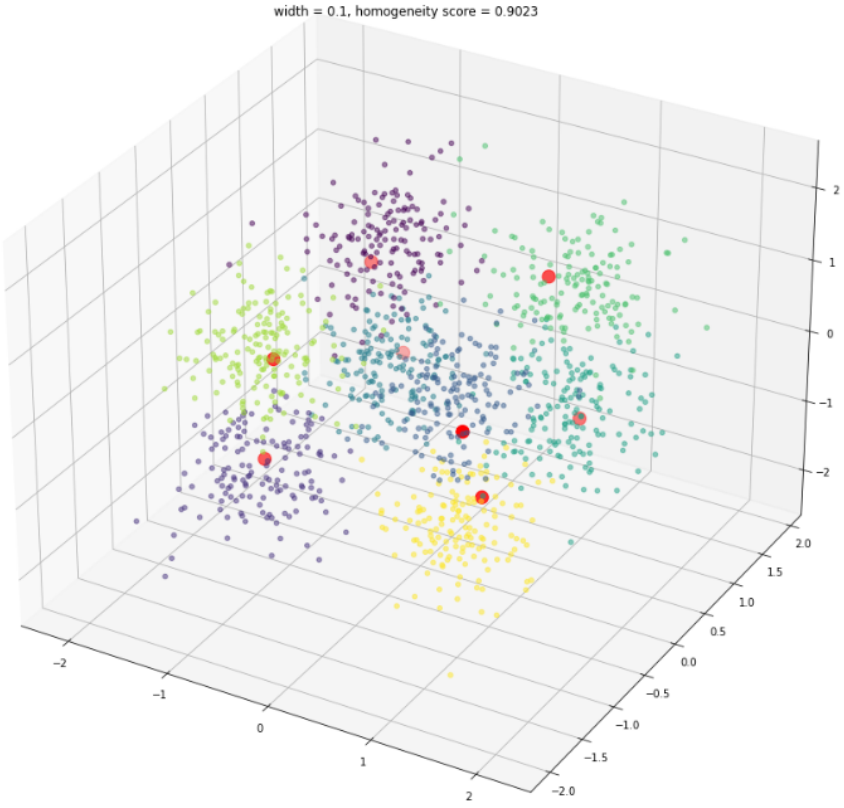
width	min dist between clusters	mean dist in clust	mean dist to clust center	silhouette
0.1	0.071119	0.851343	0.596663	0.475869
0.2	0.111860	0.851548	0.602350	0.456263
0.3	0.082029	0.848177	0.603821	0.388892
0.4	0.082029	0.896163	0.600631	0.473625
0.5	0.103118	0.852389	0.603170	0.449081
0.6	0.047695	0.852518	0.598303	0.432927
0.7	0.066884	0.852038	0.602294	0.467843
0.8	0.060737	0.848885	0.597054	0.477469
0.9	0.077468	0.852129	0.607966	0.477489
1.0	0.081513	0.860317	0.603244	0.470196

Tabela 4:

Identyczna sytuacja jak poprzednio, jedyną zmienność obserwujemy w miarze minimalnej odległości między klastrami.



Rysunek 13: klasteryzacja z parametrem width = 0.3



Rysunek 14: poprawna klasteryzacja

Praktycznie identycznie wyglądają klastry przyporządkowane przez sieć dla kolejnych wartości parametru, poza ciekawym przypadekiem dla $width = 0.3$. Zainicjowane wagi wypadły w taki sposób, że algorytm nie odnalazł wszystkich 8 wierzchołków sześcianu.

A jak z zadaniem radzą sobie różne rozmiary map?
Ponownie, nie widać szczególnych różnic jakościowych w klastrach dobranych przez sieci z różnymi architekturami.

architecture	silhouette	completeness	homogeneity	adjusted rand score
3x3	0.472980	0.892269	0.891434	0.873785
5x5	0.472169	0.896867	0.895307	0.875758
7x7	0.471407	0.900694	0.899466	0.886488
9x9	0.473005	0.892205	0.891119	0.873259
11x11	0.476888	0.906691	0.906052	0.897064
13x13	0.466475	0.895802	0.892667	0.868717

Tabela 5:

2.3.5 odpowiedzi na pytania

Czy klastry w odwzorowaniu znalezionym przez sieć pokrywają się w liczbą klastrów w faktycznych danych?
Zdecydowanie, w każdym z badanych przypadków klasteryzacja przebiegła poprawnie. Badanie przy pomocy miary silhouette potwierdziło bezsprzecznie optymalną liczbę klastrów.

Czy znalezione klastry pokrywają się z identyfikatorami wierzchołków?
Owszem, poza wyjątkowym przypadkiem width = 0.3, w każdym z badanych modeli odnaleźliśmy wierzchołki odpowiednio sześciokąta i sześcianu.

2.3.6 Podsumowanie

Obie funkcje sąsiedztwa spełniają zadanie, w przypadku funkcji gaussowskiej żadnych problemów. Funkcja kapelusza meksykańskiego, przy pewnych implementacjach miewa tendencję do propagowania wag neuronów do ∞. Przewagą kapelusza meksykańskiego nad funkcją gaussowską jest większa dyskryminacja klastrów neuronów. Wynika to z faktu, że funkcja jest bardziej lokalna, oraz ma ujemne fragmenty - to zapobiega aktywacji neuronów, które nie są blisko zwycięskiego neuronu (BMU - best matching unit), jednocześnie intensywniej oddziałując na te neurony, które są blisko BMU.

3 KOH2 - Sieć Kohonena na siatce sześciokątnej

3.1 Cel ćwiczenia

Celem, było dodanie do implementacji z poprzedniego paragrafu możliwości ułożenia neuronów w siatce sześciokątnej. Należało przeanalizować otrzymane mapowanie danych, i uwzględniając etykiety danych odpowiedzieć na pytanie **Jak dobrze znalezione klastry odpowiadają podziałowi na klasy?**

W celu analizy należało wykorzystać dwie bazy danych:

- MNIST - baza danych ręcznie pisanych cyfr (28x28 pikseli), etykietyzowana również ręcznie.
- Human Activity Recognition(HAR) Using Smartphones Data Set - baza danych zbudowana z rejestrów smartfonowych 30 osób wykonujących codzienne aktywności. Zawiera informacje w trzech wymiarach z akcelerometru, trzy-wymiarowe informacje o prędkościach kątowych z żyroskopu, dane w domenie czasu, dane w domenie częstości.

Są to wielowymiarowe dane, niemożliwe do reprezentacji graficznej bez utraty szczegółowości.

3.2 opracowanie, wnioski, wyniki

3.2.1 różnice między architekturami

Siatka sześciokątna:

- Siatki heksagonalne są często preferowane, gdy chcemy zachować relacje topologiczne między sąsiadującymi neuronami. Struktura siatki heksagonalnej przypomina naturalne wzorce występujące w różnych systemach.
- Siatki heksagonalne oferują symetrię w odniesieniu do odległości między neuronami. Odległość od dowolnego neuronu do jego sześciu najbliższych sąsiadów jest taka sama, co może być korzystne w niektórych zastosowaniach.
- Siatki heksagonalne mogą efektywniej pokrywać daną przestrzeń w porównaniu do siatek prostokątnych. Oznacza to, że dzięki siatce heksagonalnej można reprezentować większy obszar za pomocą mniejszej liczby neuronów, co potencjalnie prowadzi do bardziej zwartej sieci.

Siatka prostokątna:

- Siatki prostokątne są prostsze do implementacji i analizy w porównaniu do siatek heksagonalnych. Jednolite odstępy i regularna struktura siatki prostokątnej ułatwiają obliczenia i wykonywanie operacji.
- Jeśli dane wejściowe mają naturalną strukturę prostokątną, taką jak obrazy czy tekst, użycie siatki prostokątnej może lepiej pasować do rozkładu danych. Dopasowanie to może ułatwić interpretację wyników sieci.

3.2.2 metryki

- homogeneity: miara tego, jak bardzo klastry zawierają tylko przedstawicieli jednej grupy. Klasteryzacja spełnia wymóg homogeniczności, jeżeli każdy klaster zawiera tylko przedstawicieli pojedynczej grupy. Wartości są z przedziału od 0 do 1, gdzie 1 to idealna homogeniczność
- completeness: miara tego, jak bardzo przedstawiciele danej grupy przypisani są do tego samego klastra. Klasteryzacja spełnia wymóg komplet-

ności, jeżeli wszyscy przedstawiciele danej grupy przypisani zostaną do pojedynczego klastra. wartości jak w homogeniczności.

- V-measure[1]: średnia harmoniczna dwóch powyższych wyników, daje zbalansowaną metrykę uwzględniającą oba aspekty. wartości jak wyżej, $V - measure = 1$ oznacza idealną klasteryzację.
- adjusted rand score[2]: miara podobieństwa między etykietami a pogrupowaniem wynikającym z klasteryzacji. Rozważana jest każda para punktów danych i mierzona jest liczba takich, które:
  - należą do tej samej grupy i mają te same etykiety
  - należą do innych grup i mają różne etykiety

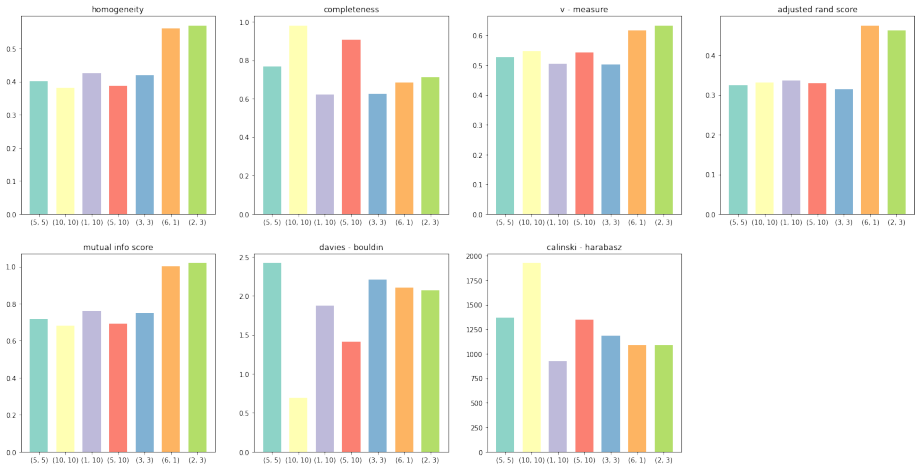
  Wartości są z zakresu od -1 do 1, gdzie 1 oznacza idealne pogrupowanie, 0 oznacza losowe grupowanie, a wartości ujemne oznaczają grupowanie gorsze, niż byśmy się spodziewali po pogrupowaniu losowym. można to interpretować w pewnym stopniu jako *ortogonalność pogrupowań*[3]
- mutual info score: mierzy ilość informacji dzieloną między grupowaniem po klasteryzacji, a etykietami i kwantyfikuje zależność między nimi.
- oraz dwie metryki nieodwołujące się do etykiet
  - Davies-Bouldin index[4]: Mierzy średnią jakość separacji między klastrami, biorąc pod uwagę szerokość klastrów i odstęp między klastrami. Im wartości są bliższe 0, tym lepsza separacja między klastrami.
  - Calinski-Hrabasz index[5]: jest to proporcja sumy rozproszenia między klastrami i wewnątrz klastrów, dla każdego klastra, gdzie rozproszenie mierzone jest jako suma kwadratów odległości.

3.2.3 Dataset HAR

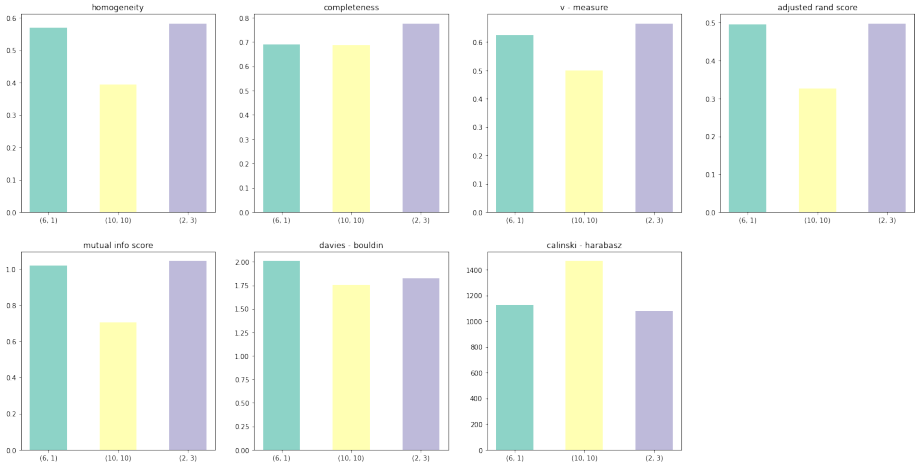
Dane podzielone są na 6 klastrów, oznaczających różne rodzaje aktywności.

Dane mają 561 wymiarów, natomiast analiza głównych składowych (PCA) wskazała, że jesteśmy w stanie uprościć dane do dwóch wymiarów zachowując przy tym zmienność na poziomie 61.6% dla osi OX, oraz 4.9% dla osi OY.

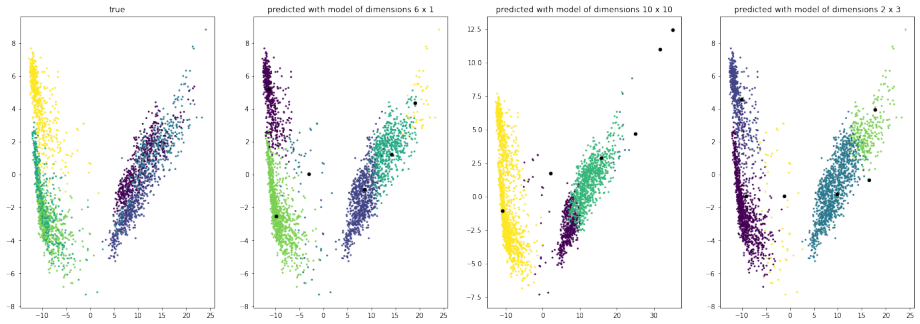
[1] https://scikit-learn.org/stable/modules/clustering.html#homogeneity-completeness-and-v-measure
[2] https://scikit-learn.org/stable/modules/clustering.html#rand-index
[3] https://stackoverflow.com/questions/42418773/how-can-we-interpret-negative-adjusted-rand-index
[4] https://scikit-learn.org/stable/modules/clustering.html#davies-bouldin-index
[5] https://scikit-learn.org/stable/modules/clustering.html#calinski-harabasz-index



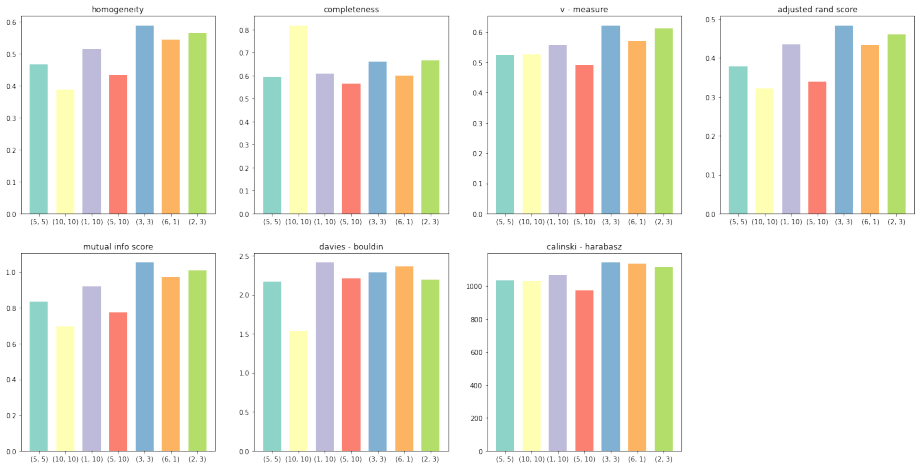
Siatka prostokątna, gaussowska funkcja sąsiedztwa powyżej, krótko trenowane modele różnych rozmiarów - celem było sprawdzenie, jakie wymiary sieci dają lepsze wyniki. Większe sieci dają lepszą kompletność, asymetryczne siatki - homogeniczność. Na podstawie tych wyników, do dłuższego trenowania zdecydowałem się poddać trzy modele, o rozmiarach 6x1, 10x10 oraz 2x3. Poniżej zaprezentowane wyniki treningu na danych testowych.



Warto spojrzeć jeszcze na PCA (na czarno zaznaczone wagi neuronów w sieci, po zmniejszeniu ich liczby do 6):

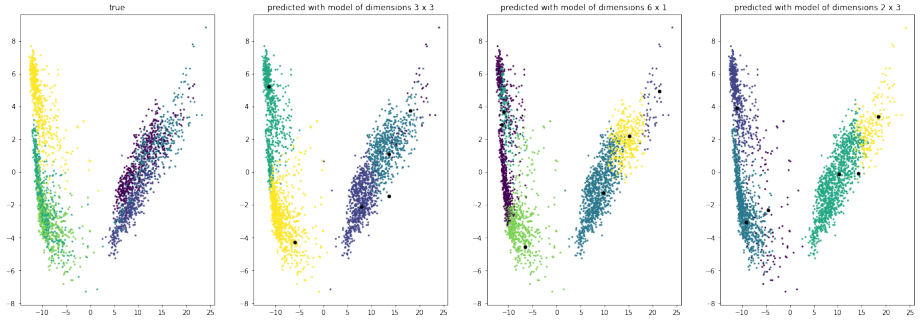


Siatka prostokątna, kapelusz meksykański jako funkcja sąsiedztwa Niezależnie od funkcji sąsiedztwa, dla tego zbioru przy architekturze prostokątnej otrzymujemy bardzo podobne wyniki przy ustalonych wymiarach sieci.

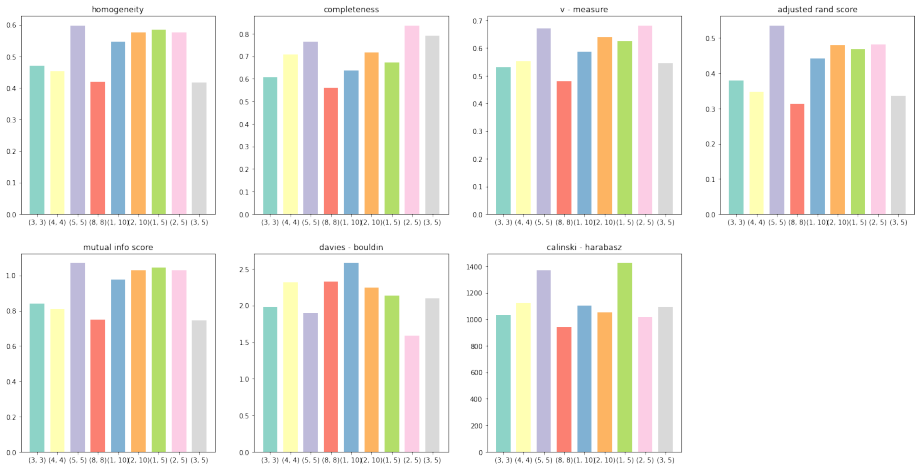


Wymiary architektur takie same jak wcześniej okazały się dawać najlepsze rezultaty, zatem w przypadku i tej funkcji aktywacji sprawdzone zostały one na większej liczbie iteracji, wyniki otrzymaliśmy bardzo zbliżone do tych, z gaussowską funkcją sąsiedztwa.

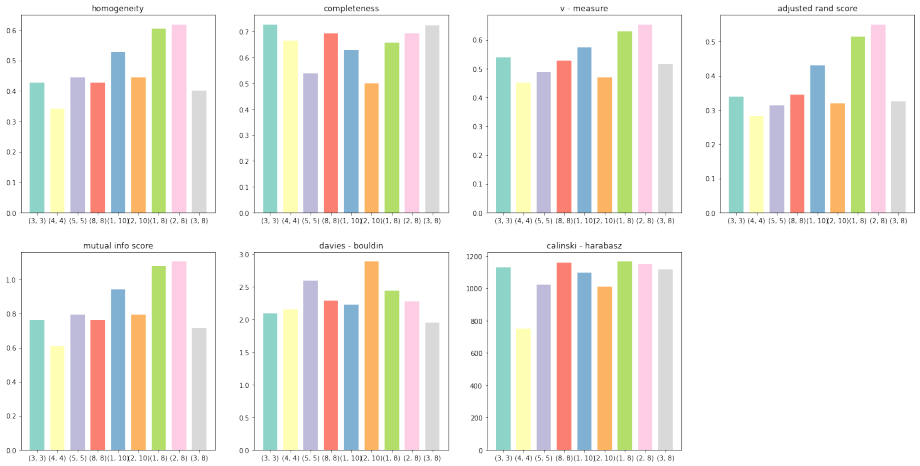
PCA dla trzech dłużej trenowanych modeli prezentuje się następująco:



Siatka sześciokątna, gaussowska funkcja sąsiedztwa: Architektura sześciokąta daje lepsze wyniki dla tego zadania.



Siatka sześciokątna, kapelusz meksykański jako funkcja sąsiedztwa: Wyniki nieco gorsze, niż w przypadku gaussowskiej funkcji sąsiedztwa

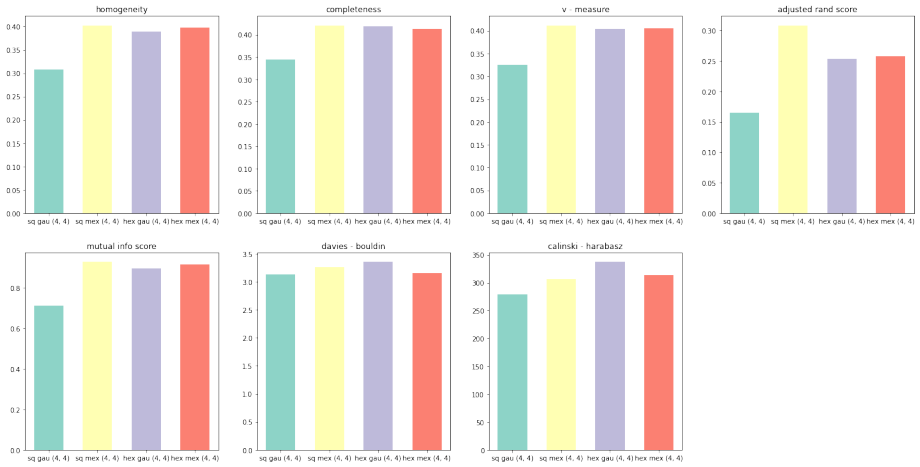


3.2.4 Dataset MNIST

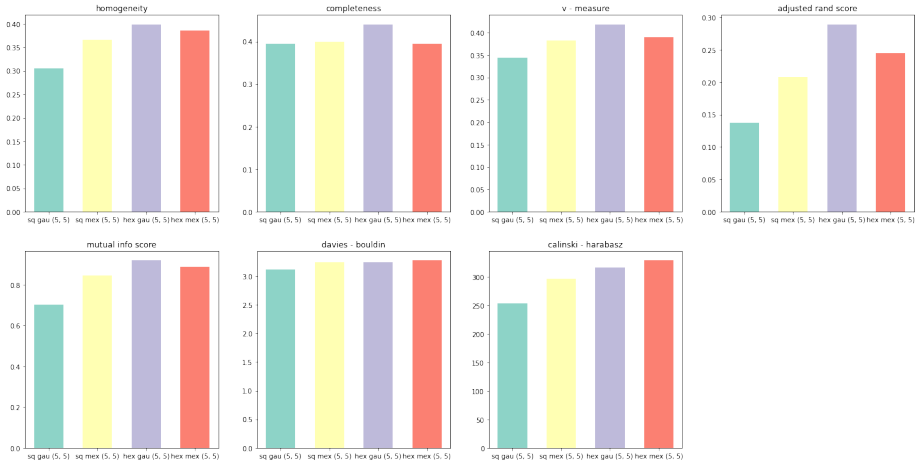
Dane zawierają 784 cechy (tyle pikseli reprezentuje obraz napisanej cyfry), klastrów jest 10.
Dwuwymiarowa reprezentacja danych przy pomocy PCA nie ma szczególnego sensu, ponieważ uproszczenie do dwóch wymiarów odejmuje ponad 80% zmienności danych

Przeprowadzając analizę jakie wymiary sieci są optymalne, okazało się że dla tego zadania najlepiej radzą sobie siatki NxN.

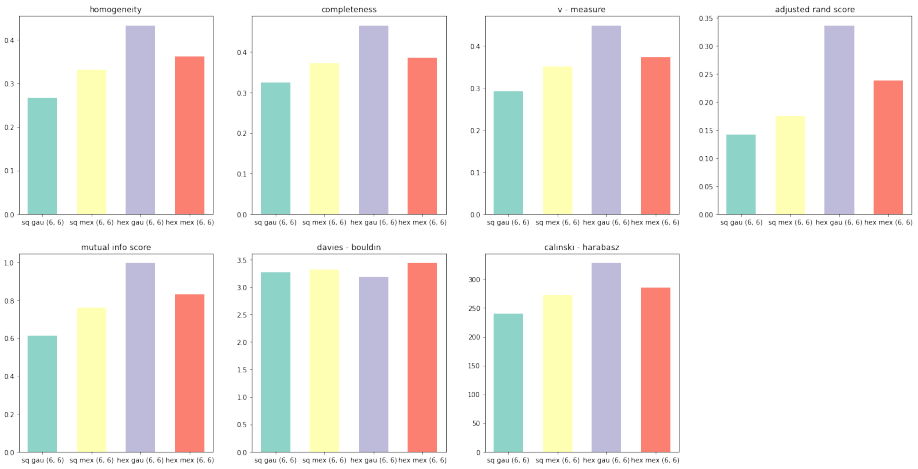
Zdecydowałem się porównać modele z siatkami NxN:



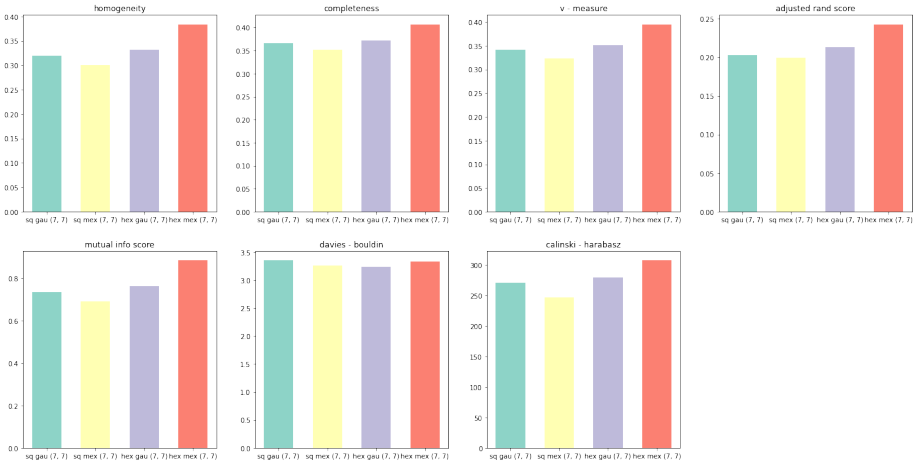
Rysunek 15: siatka 4x4



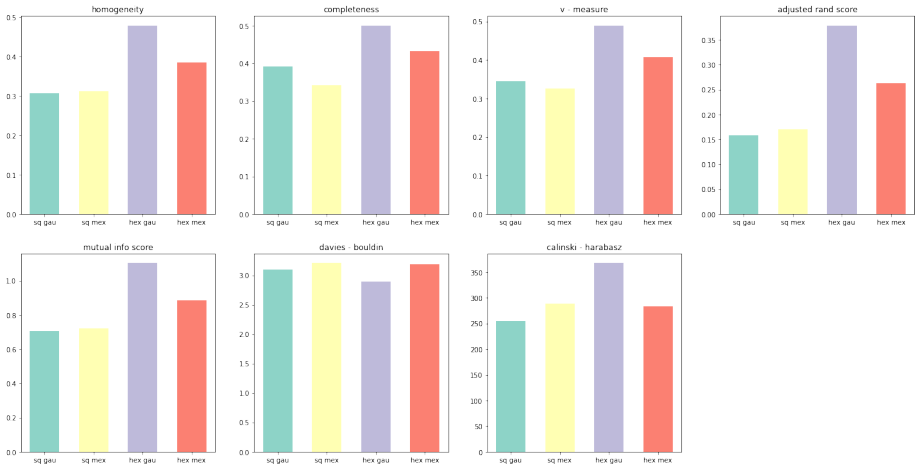
Rysunek 16: siatka 5x5



Rysunek 17: siatka 6x6



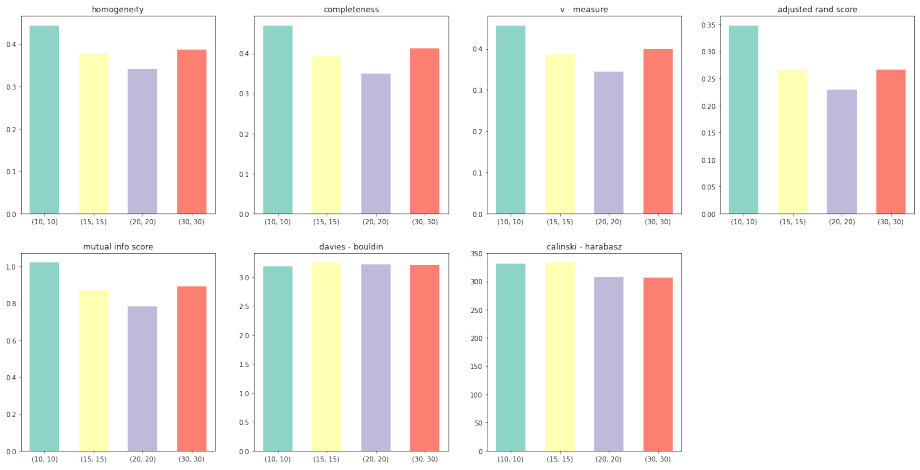
Rysunek 18: siatka 7x7



Rysunek 19: siatka 8x8

Siatki o topologii sześciokąta okazują się działać zdecydowanie lepiej dla tego zadania.

Czy w takim razie może większe, sześciokątne siatki NxN osiągną lepsze wyniki?



Rysunek 20: topologia sześciokąta, gaussowska funkcja sąsiedztwa

3.2.5 Przeanalizować otrzymane mapowanie danych uwzględniając etykiety danych. Jak dobrze znalezione klastry odpowiadają podziałowi na klasy?

Zadanie pierwsze okazało się tym prostszym, sieć była w stanie dość dobrze odwzorować mapowania z etykiet. Zadanie drugie sprawiło więcej problemów i nie osiągnięto tak dobrego wyniku jak w przypadku zadania pierwszego, natomiast wciąż istnieje pewna korelacja między dopasowanymi klastrami, a etykietami danych. Podsumowując, w żadnym z przypadków nie otrzymaliśmy idealnego dopasowania (jak to miało miejsce w przypadku pierwszej pracy domowej, gdzie odnalezione wierzchołki w prawie każdym przypadku pokrywały się z teoretycznymi), natomiast nie oznacza to, że nie udało się dopasować do danych w ogóle.