210200	210294
numer albumu	*numer albumu*
Justyna Hubert	Karol Podlewski
imię i nazwisko	*imię i nazwisko*

Data	9 maja 2018
Kierunek	Informatyka
Rok akademicki	2017/18
Semestr	IV
Grupa Dziekańska	I

Laboratorium inteligentnej analizy danych

Zadanie 2: Samoorganizująca Sieć Neuronowa

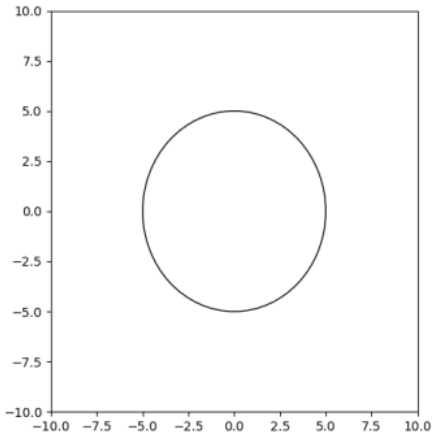
Samoorganizująca Sieć Neuronowa - wstęp

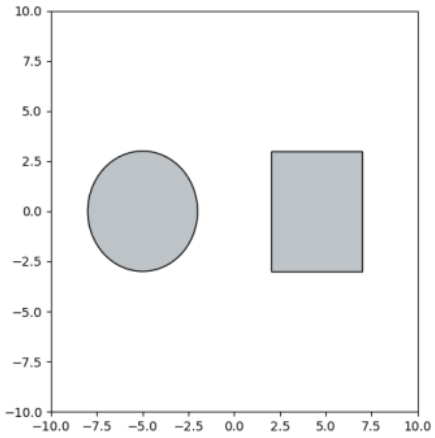
Celem ćwiczenia było stworzenie samoorganizującej się sieci neuronowej, która uczyła by się za pomocą jednego z algorytmów:

- Algorytm Kohonena,
- Algorytm gazu neuronowego,
- Algorytm K-średnich.

Sieć taka składa się ze zbioru neuronów, które program interpretuje jako punkty w przestrzeni wzorców wejściowych (danych treningowych). Neurony rywalizują za sobą o prawo do reprezentowania wzorca wejściowego, co można zrozumieć jako rywalizację o prawo do nauki.

Za każdym razem liczba danych treningowych wynosiła 222. W przypadku dwóch figur rozbijana była po równo na obie figury. W przypadku jednej figury dane treningowe zawsze były umieszczane w okręgu o środku w punkcie (0,0) oraz promieniu 5, zaś w przypadku dwóch figur było to koło o środku w punkcie (-5,0) i promieniu 3 oraz w prostokącie o skrajnych narożnikach w punktach (2,3) oraz (7,-3). Figury te przedstawione na wykresie wyglądały następująco:





Przyjęte oznaczenia

W sprawozdaniu będzie się można natknąć na następujące skróty:

- BK – błąd kwantyzacji
- OS – odchylenie standardowe
- LNN – liczba nieaktywnych neuronów
- WS – współczynnik sąsiedztwa
- Inic1 – neurony zostały zainicjalizowane w równej odległości na linii na środku wykresu
- Inic2 – neurony zostały zainicjalizowane w równej odległości na dwóch, symetrycznych pionowych liniach
- MOL – mniejszy obszar na którym losowane były początkowe położenia neuronów

Męczenie się neuronów

Męczenie się neuronów ma zapobiec ciągłemu wygrywaniu jednego neuronu w przypadku korzystnego wylosowania położenia względem innych neuronów. Zdecydowaliśmy się na wprowadzenie potencjału dla każdego neuronu, który jest modyfikowany przy każdym znalezieniu zwycięskiego neuronu. Zakładając zwycięstwo w-tego neuronu, potencjał obliczamy ze wzoru:

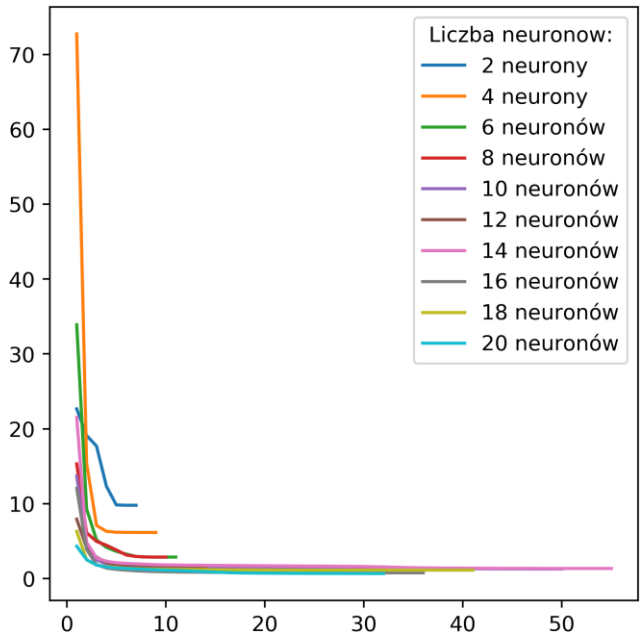
$$p_i(k+1) = \begin{cases} p_i(k) + \frac{1}{n} & \text{dla } i \neq w \\ p_i(k) - p_{\min} & \text{dla } i = w \end{cases}$$

Współczynnik p_{min} oznacza minimalny potencjał upoważniający do udziału we współzawodnictwie. Jeśli aktualna wartość potencjału p_i spadnie poniżej p_{min} to neuron i-ty „odpoczywa", a zwycięzcy poszukuje się spośród neuronów, które mają odpowiedni potencjał. Maksymalną wartość potencjału ograniczamy na 1, zaś wartość p_{min} ustaliliśmy na poziomie 0,75 w przypadku Kohonena oraz 0,9 w przypadku algorytmu Centroidów.

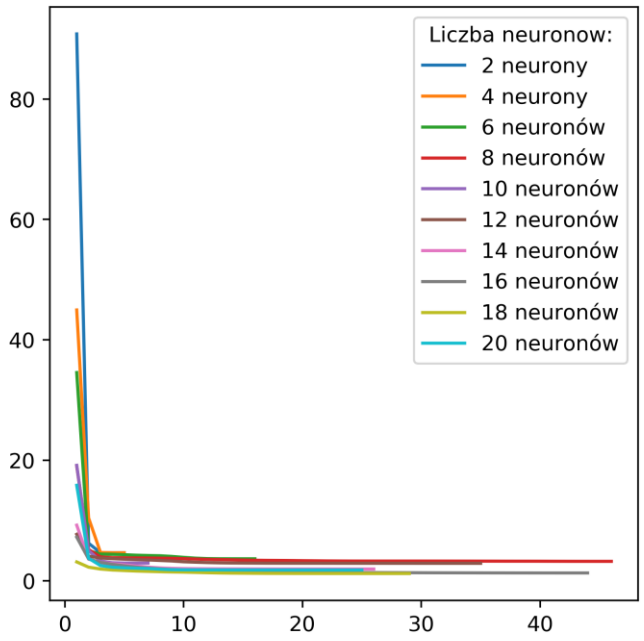
1. Algorytm Kohonena

Zastosowany przez nas algorytm zakłada sąsiedztwo Gaussowskie.

Na początku sprawdziliśmy jak zmienia się błąd kwantyzacji dla różnej liczby neuronów, których początkowe położenie jest losowane, dla kroku nauki 0.02 i braku sąsiedztwa:



Rys1. Zmiana błędu kwantyzacji dla 1 figury



Rys2. Zmiana błędu kwantyzacji dla 2 figur

W zależności od początkowego losowania wag błąd może być bardzo duży, co najlepiej widać dla małej ilości wag, gdzie szansa na niewylosowanie początkowego położenia neuronu w bliskiej odległości od danych treningowych jest wyższa. Błąd ten szybko spada, i zazwyczaj przy mniejszej ilości neuronów oraz „gorszym" rozłożeniu punktów treningowych będzie on na koniec wyższy, niż w odwrotnej sytuacji. Zbyt duża liczba neuronów nie zawsze powoduje osiągnięcie optymalnego wyniku błędu kwantyzacji, jednak są to minimalne różnice.

Kolejną rzeczą było sprawdzenie wpływu kroku nauki, sąsiedztwa, zastosowania mechanizmu męczenia się neuronów oraz sposobu inicjalizacji wag na osiągnięty błąd kwantyzacji czy liczbę nieaktywnych neuronów. Nauka była przerywana po 150 iteracjach. Sieć w obu przypadkach miała 20 neuronów:

Tab1. Wyniki nauki dla sieci z jedną figurą geometryczną

Inicjalizacja	Męczenie	Krok nauki	WS	Śr. BK	Min. BK	OS BK	Śr. LNN	Min. LNN	OS śr. LNN
Losowa	-	0,01	0,1	0,93782	0,61837	0,17932	14	7	1,51440
Losowa	-	0,03	0,3	0,56039	0,45619	0,06958	7	0	1,28900
Losowa	Jest	0,03	0,3	0,57387	0,48139	0,05079	1	0	0,82003
Inic1	Jest	0,03	0,3	0,54811	0,49691	0,02741	3	0	2,26980
Inic2	Jest	0,03	0,3	0,54289	0,48284	0,03922	5	0	1,76182

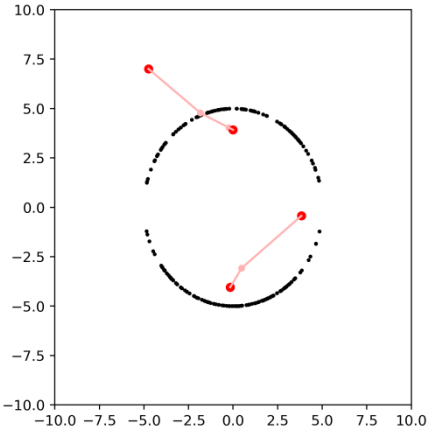
Tab2. Wyniki nauki dla sieci z dwiema figurami geometrycznymi

Inicjalizacja	Męczenie	Krok nauki	WS	Śr. BK	Min. BK	OS BK	Śr. LNN	Min. LNN	OS śr. LNN
Losowa	-	0,01	0,1	1,04540	0,89392	0,08510	12	4	1,61048
Losowa	-	0,03	0,3	0,84308	0,72945	0,06615	4	0	2,01336
Losowa	Jest	0,03	0,3	1,45614	1,22976	0,23832	0	0	0,13263
Inic1	Jest	0,03	0,3	1,39075	1,24569	0,07211	0	0	0,11011
Inic2	Jest	0,03	0,3	1,34327	1,22936	0,05543	0	0	0,18963

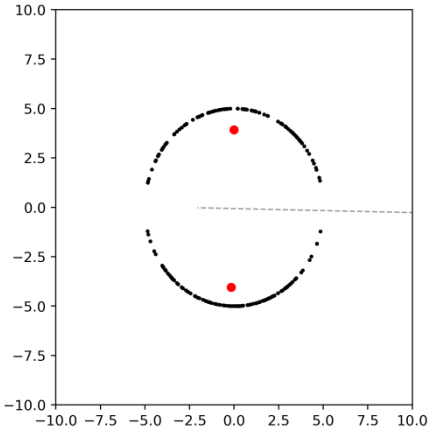
Pozytywny wpływ na naukę sieci miało zwiększenie kroku nauki jak i współczynnika sąsiedztwa. Błąd kwantyzacji był mniejszy, a średnia liczba nieaktywnych neuronów mniejsza. Należy jednak pamiętać o umiarze, gdyż zbyt wysokie parametry potrafią niemiłosiernie wydłużyć naukę bądź nawet sprawić, że sieć nigdy nie znajdzie odpowiedniego położenia neuronów. Inicjalizacja neuronów w uporządkowany sposób potrafiła zarówno poprawić jak i pogorszyć naukę, co oczywiście zależy od miejsca w którym zainicjujemy neurony względem tego gdzie znajdują się dane treningowe – im bliższe sobie będą, tym neurony lepiej będą się uczyć.

Wprowadzenie męczenia się neuronów znacząco zmniejsza liczbę nieaktywnych neuronów, jednak błąd kwantyzacji potrafi się zwiększyć. Więcej neuronów się uczy, jednak nauka konkretnych zwycięzców potrafi być przez to wolniejsza, przez co nie osiągną tak dobrej dokładności.

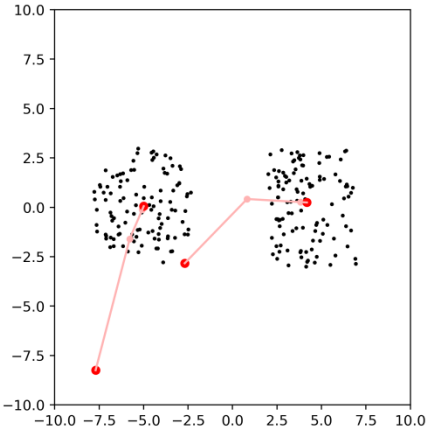
Ostatnią rzeczą jaką chcieliśmy sprawdzić w przypadku algorytmu Kohonena była historia nauki oraz końcowe przypisanie neuronów do danych treningowych. Dzięki wcześniejszym eksperymentom wiemy, że w przypadku 2 neuronów sieć będzie się lepiej uczyć przy metodzie WTA, zaś w przypadku 10 neuronów przyda się zastosować sąsiedztwo:



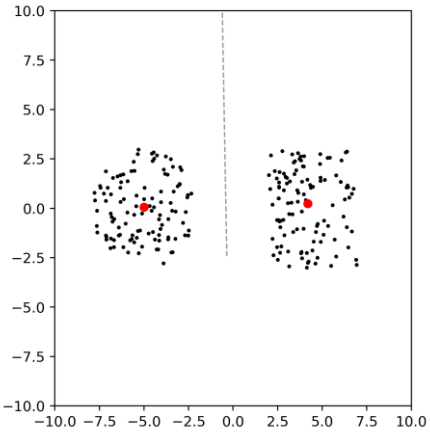
Rys3. Historia nauki dla 2 neuronów z 1 figurą geometryczną



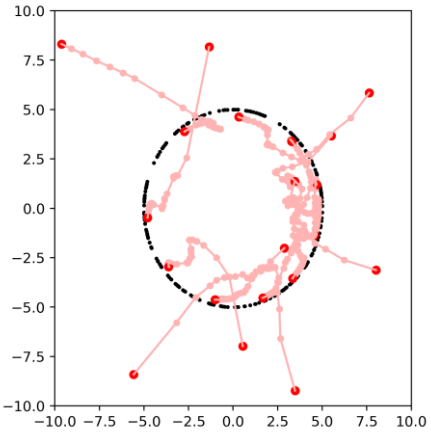
Rys4. Diagram Vornoia dla 2 neuronów z 1 figurą geometryczną



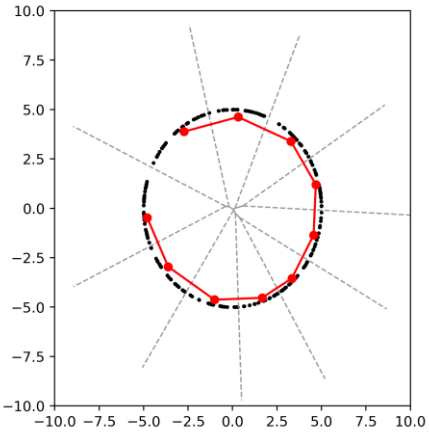
Rys5. Historia nauki dla 2 neuronów z 2 figurami geometrycznymi



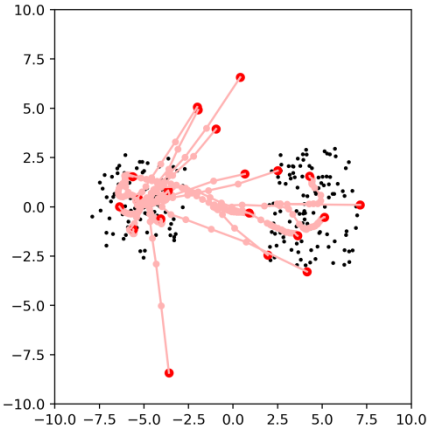
Rys6. Diagram Vornoia dla 2 neuronów z 2 figurami geometryczną



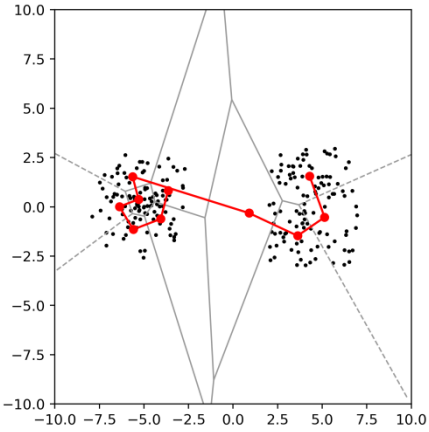
Rys7. Historia nauki dla 10 neuronów z 1 figurą geometryczną



Rys8. Diagram Vornoia dla 10 neuronów z 1 figurą geometryczną



Rys9. Historia nauki dla 10 neuronów z 2 figurami geometrycznymi

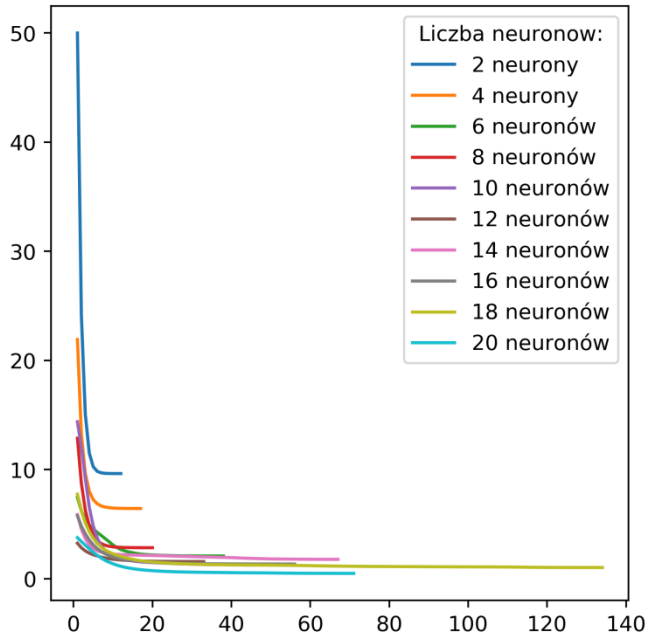


Rys10. Diagram Vornoia dla 10 neuronów z 2 figurami geometryczną

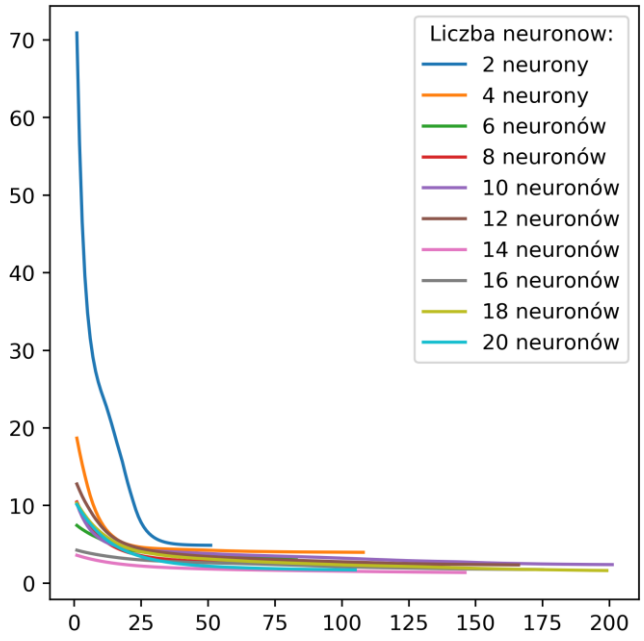
Z diagramów Vornoia może wynikać wniosek, że nie wszystkie neurony się uczą, co jest nie prawdą. Dzięki zastosowaniu sąsiedztwa wszystkie neurony się uczą, w dodatku w przypadku sąsiedztwa prostokątnego neurony będą ułożone po kolei.

2. Algorytm gazu neuronowego

Na początku sprawdziliśmy jak zmienia się błąd kwantyzacji dla różnej liczby neuronów. Sieć była zainicjalizowana z krokiem nauki 0,005 oraz współczynnikiem sąsiedztwa 0,01:



Rys11. Zmiana błędu kwantyzacji dla 1 figury



Rys12. Zmiana błędu kwantyzacji dla 2 figur

Wnioski są identyczne z tymi dla algorytmu Kohonena, dlatego nie będziemy się powtarzać.

Kolejną rzeczą było sprawdzenie wpływu kroku nauki, sąsiedztwa, zastosowania mechanizmu męczenia się neuronów oraz sposobu inicjalizacji wag na osiągnięty błąd kwantyzacji czy liczbę nieaktywnych neuronów. Sieć przestawała się uczyć po 150 iteracjach. W obu przypadkach miała 20 neuronów:

Tab3. Wyniki nauki dla sieci z jedną figurą geometryczną

Inicjalizacja	Krok nauki	WS	Śr. BK	Min. BK	OS BK	Śr. LNN	Min. LNN	OS śr. LNN
Losowa	0,005	0,01	0,89164	0,70370	0,12267	11	9	1,06467
Losowa	0,05	0,1	0,94397	0,69910	0,15210	10	1	1,72375
Losowa	Wagi ze wzoru*		0,43138	0,39070	0,02376	2	0	0,86219
Inic1	Wagi ze wzoru*		0,39229	0,35076	0,02533	1	0	0,52678
Inic2	Wagi ze wzoru*		0,36443	0,34672	0,01179	0	0	0,00000

Tab4. Wyniki nauki dla sieci z dwiema figurami geometrycznymi

Inicjalizacja	Krok nauki	WS	Śr. BK	Min. BK	OS BK	Śr. LNN	Min. LNN	OS śr. LNN
Losowa	0,001	0,01	1,18952	1,02363	0,09019	9	5	1,38651
Losowa	0,01	0,1	1,04885	0,93792	0,09823	4	0	1,35185
Losowa	Wagi ze wzoru*		0,78067	0,71702	0,03272	1	0	0,53026
Inic1	Wagi ze wzoru*		0,76409	0,72267	0,02350	0	0	0,44937
Inic2	Wagi ze wzoru*		0,76470	0,71735	0,02462	0	0	0,32532

* krok nauki wyliczany był ze wzoru:

$$\eta = 0{,}05\left(\frac{0{,}003}{0{,}05}\right)^{\frac{iteracja}{20}}$$

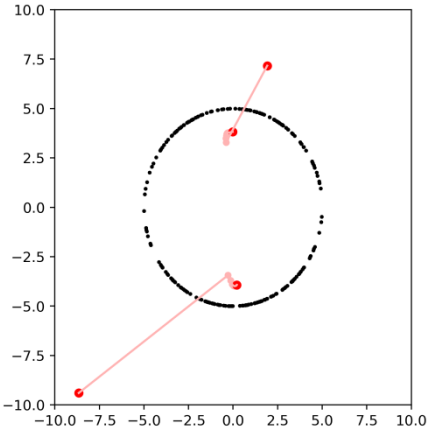
współczynnik sąsiedztwa wyliczany był ze wzoru:

$$\lambda = \frac{ilość\_neuronów}{4}\left(\frac{2 \cdot 0{,}01}{ilość\_neuronów}\right)^{\frac{iteracja}{20}}$$

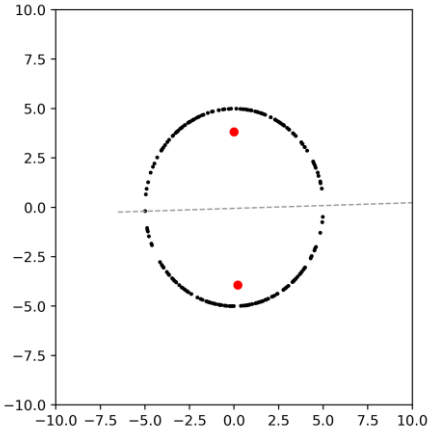
Pozytywny wpływ na naukę sieci miało zwiększenie kroku nauki, zdecydowanie mniejszy efekt dawała zmiana współczynnika sąsiedztwa. W przypadku inicjalizacji neuronów w konkretnych miejscach widać pozytywne efekty zarówno w błędzie średniokwadratowym jak i licznie nieaktywnych współczynników.

Rewelacyjny wpływ na działanie sieci ma obliczanie współczynników na bieżąco. Dzięki ciągłej zmianie współczynników neurony są w stanie szybko rozpocząć naukę, a będąc coraz bliżej celu przygotować się na delikatniejsze zmiany. Ze względu na działanie algorytmu trudno było tu zastosować męczenie się neuronów, jednak nie było ono potrzebne.

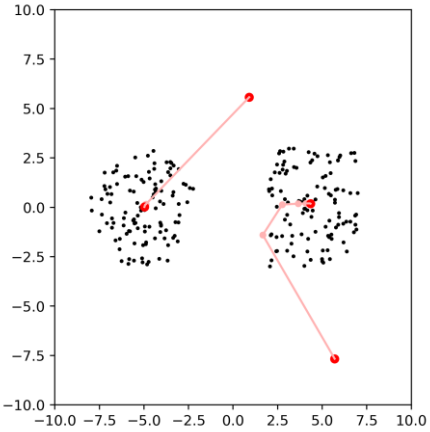
Ostatnią rzeczą jaką chcieliśmy sprawdzić w przypadku algorytmu gazu neuronowego była historia nauki oraz końcowe przypisanie neuronów do danych treningowych. Zarówno krok nauki jak i współczynnik sąsiedztwa obliczany był na bieżąco, a początkowe położenie neuronów było losowe.



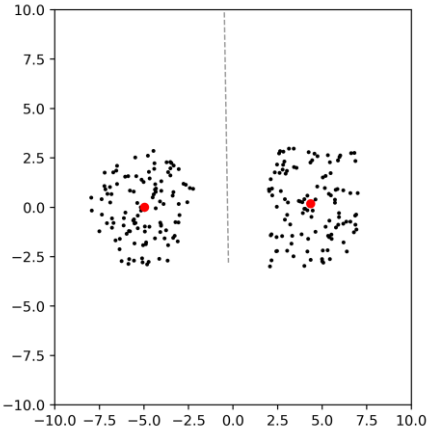
Rys13. Historia nauki dla 2 neuronów z 1 figurą geometryczną



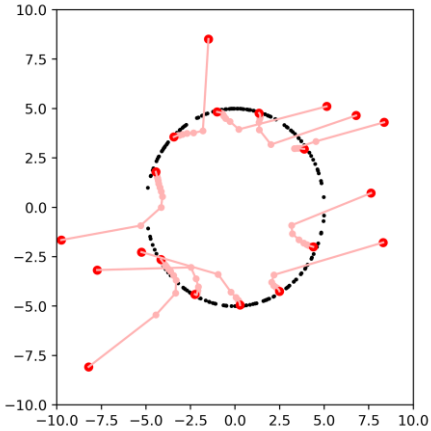
Rys14. Diagram Vornoia dla 2 neuronów z 1 figurą geometryczną



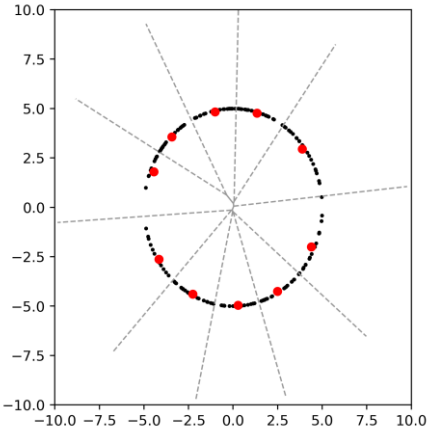
Rys15. Historia nauki dla 2 neuronów z 2 figurami geometrycznymi



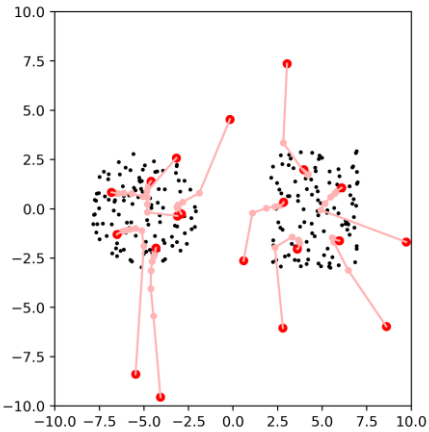
Rys16. Diagram Vornoia dla 2 neuronów z 2 figurami geometryczną



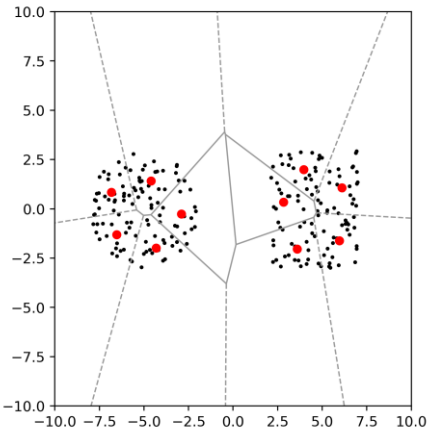
Rys17. Historia nauki dla 10 neuronów z 1 figurą geometryczną



Rys18. Diagram Vornoia dla 10 neuronów z 1 figurą geometryczną



Rys19. Historia nauki dla 10 neuronów z 2 figurami geometrycznymi

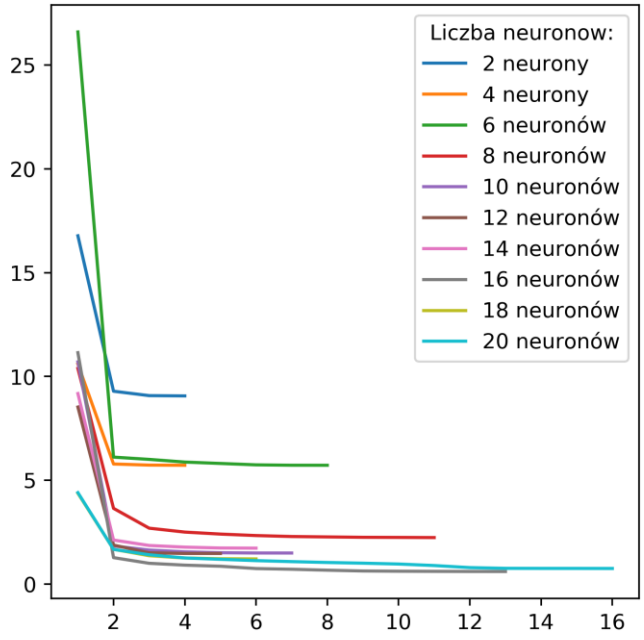


Rys20. Diagram Vornoia dla 10 neuronów z 2 figurami geometryczną

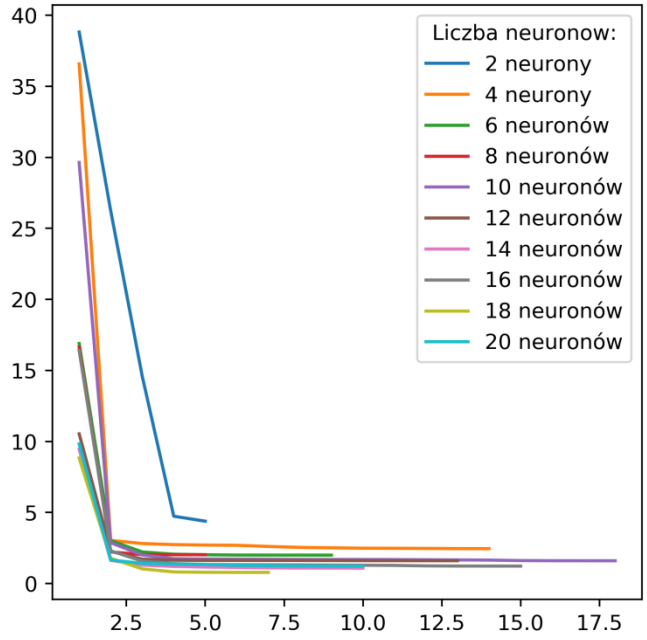
Na rysunkach doskonale widać, że neurony dosyć szybko zbliżają się do danych treningowych, po czym mniejszymi kroczkami zaczynają dążyć w stronę celu. Ze względu na sposób działania algorytmu gazu neuronowego, większość neuronów chociaż przez chwilę się uczy.

3. Algorytm K-średnich (Centroidów):

Na początku sprawdziliśmy jak zmienia się błąd kwantyzacji dla różnej liczby neuronów:



Rys21. Zmiana błędu kwantyzacji dla 1 figury



Rys22. Zmiana błędu kwantyzacji dla 2 figur

Wnioski znowu są identyczne, jak w przypadku poprzednich algorytmów. Tutaj jednak lepiej widać, że sieć z mniejszą liczbą neuronów dostosowuje się słabiej, a początkowy błąd przy większej ilości wag często jest mniejszy – choć nie zawsze. Zdecydowanie lepiej widać też, że zbyt duża liczba neuronów nie zawsze jest optymalna.

Naszą kolejną czynnością było sprawdzenie wpływu zastosowania mechanizmu męczenia się neuronów oraz sposobu inicjalizacji wag na osiągnięty błąd kwantyzacji czy liczbę nieaktywnych neuronów. Sieć przestawała się uczyć maksymalnie po 150 iteracjach. W obu przypadkach miała 20 neuronów:

Tab5. Wyniki nauki dla sieci z jedną figurą geometryczną

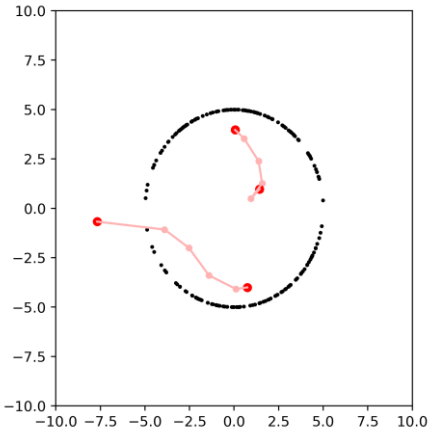
Inicjalizacja	Męczenie	Śr. BK	Min. BK	OS BK	Śr. LNN	Min. LNN	OS śr. LNN
Losowa	-	0,82902	0,64235	0,10573	15	8	0,55411
MOL	-	0,40178	0,26750	0,08862	14	4	1,04604
Inic1	-	0,36288	0,22823	0,07496	12	0	1,31512
Inic2	-	0,20551	0,16056	0,02475	13	0	1,37021
Losowa	Jest	0,87848	0,70396	0,14670	8	0	3,48480
MOL	Jest	0,74038	0,65047	0,04792	9	0	4,02599
Inic1	Jest	0,76097	0,67467	0,05701	9	0	4,86502
Inic2	Jest	0,73663	0,64120	0,06006	9	0	3,98049

Tab6. Wyniki nauki dla sieci z dwiema figurami geometrycznymi

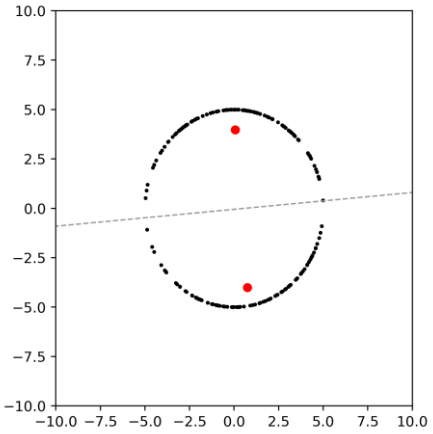
Inicjalizacja Krok nauki	Męczenie	Śr. BK	Min. BK	OS BK	Śr. LNN	Min. LNN	OS śr. LNN
Losowa	-	1,02637	0,88744	0,08664	13	5	1,15106
MOL	-	0,74897	0,56834	0,12159	13	3	1,42744
Inic1	-	1,01322	0,79798	0,12116	14	8	0,78055
Inic2	-	0,62953	0,53201	0,06739	12	4	1,01586
Losowa	Jest	1,94493	1,32253	0,25202	3	0	1,44973
MOL	Jest	1,97261	1,50721	0,18057	3	0	2,30035
Inic1	Jest	2,13746	1,70300	0,12763	2	0	0,78349
Inic2	Jest	2,17391	1,74444	0,12604	2	0	1,23402

Z wcześniejszych eksperymentów wiemy, że dla algorytmu Centroidów bez zastosowania mechanizmu męczenia początkowe położenie ma duże znaczenie, jednak przy zastosowaniu mechanizmu męczenia się neuronów, sieć uczy się porównywalnie dobrze zarówno z odpowiednim rozłożeniem neuronów, jak i jego braku – w wypadku tego algorytmu jest to spodziewany wynik. Przez zastosowanie mechanizmu zmęczenia błąd kwantyzacji jest większy, za to liczba nieaktywnych neuronów drastycznie spada.

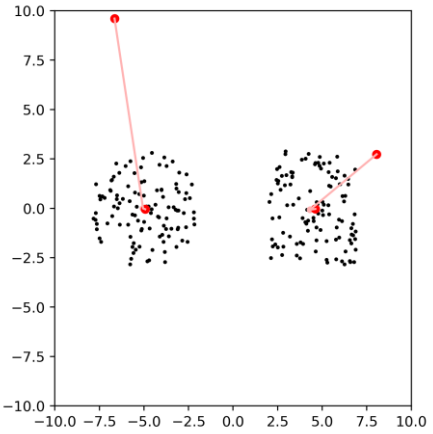
Ostatnią rzeczą jaką chcieliśmy sprawdzić w przypadku algorytmu K-Średnich była historia nauki oraz końcowe przypisanie neuronów do danych treningowych. Początkowe położenie neuronów było losowe.



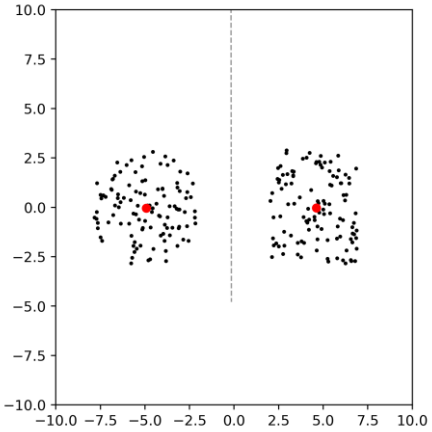
Rys23. Historia nauki dla 2 neuronów z 1 figurą geometryczną



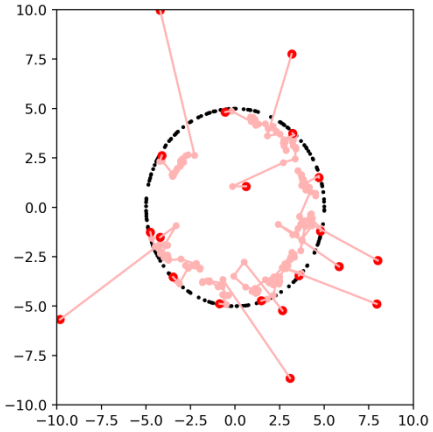
Rys24. Diagram Vornoia dla 2 neuronów z 1 figurą geometryczną



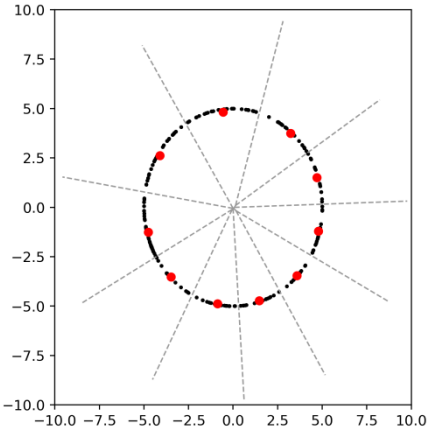
Rys25. Historia nauki dla 2 neuronów z 2 figurami geometrycznymi



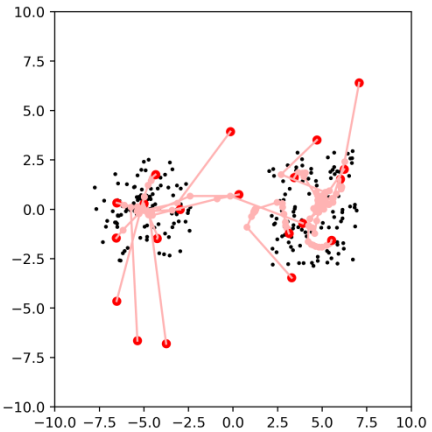
Rys26. Diagram Vornoia dla 2 neuronów z 2 figurami geometryczną



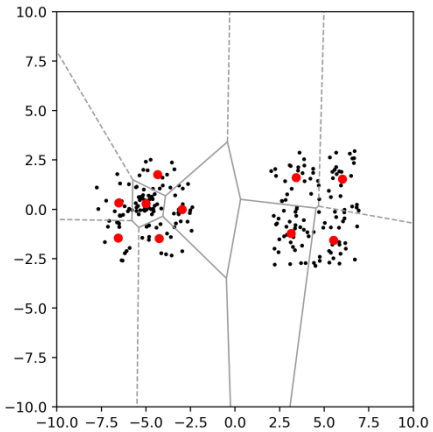
Rys27. Historia nauki dla 10 neuronów z 1 figurą geometryczną



Rys28. Diagram Vornoia dla 10 neuronów z 1 figurą geometryczną



Rys29. Historia nauki dla 10 neuronów z 2 figurami geometrycznymi



Rys30. Diagram Vornoia dla 10 neuronów z 2 figurami geometryczną

Na rysunkach wyraźnie widać, że nie wszystkie neurony biorą udział w kwantyzacji zbioru danych. Dobrym pomysłem było by dodanie mechanizmu zmęczenia, który pomógł by w nauce neuronom zbyt oddalonym od danych treningowych by w ogóle uczestniczyć w kwantyzacji. W przypadku algorytmu K-średnich dużo większe znaczenie ma początkowe położenie neuronów. Jest to świetny przykład metody nauki WTA, gdzie tylko zwycięzca się uczy.