

1 Podstawy programowania sieci neuronowych w programie Matlab 7.0

1.1 Wczytanie danych wejściowych

Pomocny przy tym będzie program Microsoft Excel. W programie tym obrabiamy wstępnie nasze dane poprzez ich skalowanie do wartości odpowiednich dla sztucznej sieci neuronowej.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
2471	2531,000	318,800	0,000	10,428	81,720	74,400	697,200	345,900	346,200	41,590	73,900	696,700	245,500	245,800	8,670
2472	2531,000	318,800	0,000	10,534	79,960	74,400	697,200	345,900	346,200	41,590	73,900	697,200	246,300	245,800	8,660
2473	2531,000	319,600	0,000	10,603	81,720	74,400	697,200	345,900	346,200	40,720	73,900	697,200	246,300	245,500	8,700
2474	2531,000	319,600	0,000	10,378	83,460	74,400	697,200	345,900	346,200	45,090	73,900	697,200	246,300	245,500	8,700
2475	2538,000	312,500	0,000	10,409	84,380	74,400	697,200	345,900	346,200	46,090	73,900	697,200	246,300	246,300	8,660
2476	2531,000	323,600	0,000	10,516	80,840	74,400	697,200	345,900	346,200	46,960	73,900	697,200	246,300	245,300	8,660
2477	2524,000	316,400	0,000	10,525	82,590	74,400	697,200	345,900	346,200	48,720	73,900	697,200	247,000	246,000	8,690
2478	2524,000	312,500	0,000	10,438	81,810	74,090	697,200	345,900	346,200	47,840	73,900	696,700	246,600	246,000	8,670
2479	2524,000	323,100	0,000	10,390	80,840	74,090	697,200	345,900	346,200	46,090	73,900	695,800	246,600	246,000	8,720
2480	2524,000	323,100	0,000	10,350	80,840	74,090	697,200	345,900	346,200	46,090	73,900	695,300	246,800	246,000	8,760
2481	2518,000	323,100	0,000	10,300	81,720	74,090	697,200	345,900	346,200	45,090	73,900	697,200	246,000	245,300	8,780
2482	2518,000	317,300	0,000	10,253	83,590	74,090	697,200	345,900	346,200	46,960	73,900	697,200	246,000	245,500	8,740
2483	2518,000	318,800	0,000	10,300	83,460	74,090	697,200	345,900	346,200	47,840	73,900	697,200	246,000	246,300	8,650
2484	2518,000	315,800	0,000	10,300	82,590	74,090	697,200	345,900	346,200	46,960	73,900	697,200	246,000	245,300	8,700
2485	2524,000	320,300	0,000	10,234	82,680	74,090	697,200	345,900	346,200	46,960	73,900	697,200	246,000	246,000	8,620
2486	2524,000	313,400	0,000	10,244	83,590	74,090	697,200	345,900	346,200	46,090	73,900	697,200	246,000	246,000	8,720
2487	2524,000	313,400	0,000	10,203	81,810	74,090	697,200	345,900	346,200	45,090	73,900	697,200	246,000	246,000	8,720
2488	2524,000	318,800	0,000	10,253	81,810	74,090	697,200	345,900	346,200	43,340	73,620	697,200	246,000	246,000	8,670
2489	2524,000	316,400	0,000	10,175	82,680	74,090	697,200	345,900	346,200	42,460	73,620	697,200	246,000	246,000	8,720
2490	2524,000	324,200	0,000	10,184	81,810	74,090	697,200	345,900	346,200	43,340	73,620	697,200	246,000	245,300	8,670
2491	2524,000	316,400	0,000	10,194	80,060	74,090	697,200	345,900	346,200	45,090	73,620	696,200	246,800	246,000	8,640
2492	2524,000	313,900	0,000	10,203	81,810	74,090	697,200	345,900	346,200	45,090	73,620	696,700	247,000	245,300	8,700
2493	2524,000	321,700	0,000	10,234	84,380	74,090	697,200	345,900	346,200	49,680	73,620	695,300	247,000	246,300	8,690
2494	2531,000	321,700	0,000	10,244	85,250	74,090	697,200	345,900	346,200	46,960	73,620	697,200	246,300	246,000	8,750
2495	2524,000	313,900	0,000	10,290	83,590	74,090	697,200	345,900	346,200	43,340	73,620	697,600	245,300	246,000	8,740
2496	2524,000	320,300	0,000	10,262	86,120	74,090	697,200	345,900	346,200	43,340	73,620	696,200	246,000	246,000	8,760
2497	2524,000	317,300	0,000	10,222	80,840	74,090	697,200	345,900	346,200	46,090	73,620	696,200	246,000	246,000	8,740
2498	2524,000	324,200	0,000	10,194	84,380	74,090	697,200	345,900	346,200	45,090	73,620	695,800	246,000	246,000	8,720
2499	2524,000	313,900	0,000	10,222	82,590	74,090	697,200	345,900	346,200	46,090	73,620	697,200	246,000	246,000	8,720
2500	2524,000	324,600	0,000	10,244	83,590	74,090	697,200	345,900	346,200	47,060	73,620	697,200	246,000	245,000	8,660
2501	2524,000	321,200	0,000	10,322	83,590	74,090	697,200	345,900	345,900	47,060	73,620	697,200	246,000	246,300	8,600
2502															
2503															
2504	2674,000	325,200	51,480	12,000	118,840	75,560	699,600	371,800	372,000	86,590	74,780	698,600	320,800	320,500	16,240
2505															

1. Tabela danych wejściowych przed formatowaniem

Wykonujemy to przez podzielenie kolumn wejściowych i wyjściowych przez element maksymalny danej kolumny.

Microsoft Excel - Grupa2Sztuczna inteligencja

Wpisz pytanie do Pomocy

Arial 10

A2501 =trening!A2501/trening!A\$2504

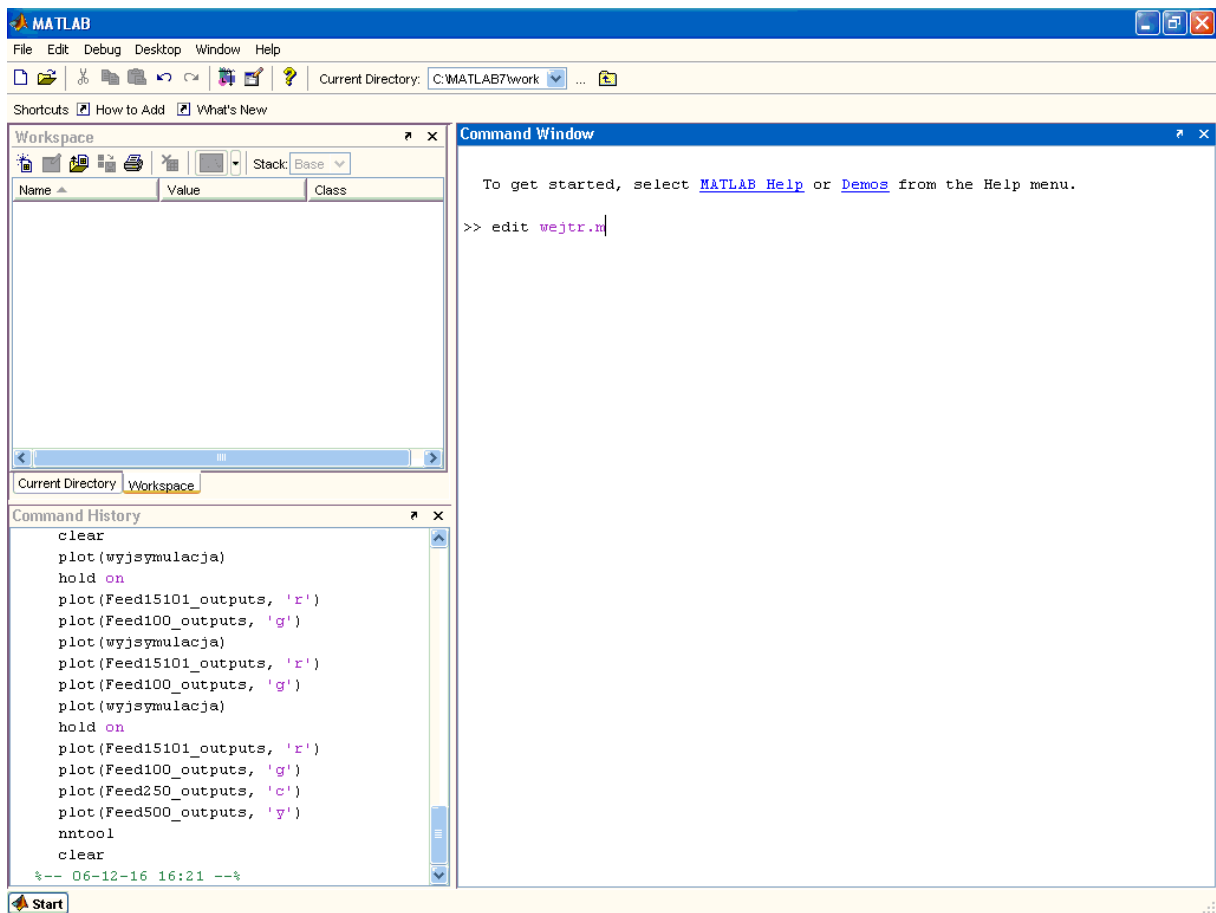
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
2478	0,943904	0,960947	0	0,869833	0,688405	0,980545	0,996569	0,930339	0,930645	0,552489	0,988232	0,99728	0,768703	0,767551	0,533867
2479	0,943904	0,993542	0	0,865833	0,680242	0,980545	0,996569	0,930339	0,930645	0,532279	0,988232	0,995992	0,768703	0,767551	0,536946
2480	0,943904	0,993542	0	0,8625	0,680242	0,980545	0,996569	0,930339	0,930645	0,532279	0,988232	0,995276	0,769327	0,767551	0,54064
2481	0,94166	0,993542	0	0,858333	0,687647	0,980545	0,996569	0,930339	0,930645	0,52073	0,988232	0,997996	0,766833	0,765367	0,54064
2482	0,94166	0,975707	0	0,854417	0,703383	0,980545	0,996569	0,930339	0,930645	0,542326	0,988232	0,997996	0,766833	0,765991	0,538177
2483	0,94166	0,98032	0	0,858333	0,702289	0,980545	0,996569	0,930339	0,930645	0,552489	0,988232	0,997996	0,766833	0,766487	0,532635
2484	0,94166	0,971095	0	0,858333	0,694968	0,980545	0,996569	0,930339	0,930645	0,542326	0,988232	0,997996	0,766833	0,765367	0,535714
2485	0,943904	0,984932	0	0,852833	0,695725	0,980545	0,996569	0,930339	0,930645	0,542326	0,988232	0,997996	0,766833	0,767551	0,530788
2486	0,943904	0,963715	0	0,853667	0,703383	0,980545	0,996569	0,930339	0,930645	0,532279	0,988232	0,997996	0,766833	0,767551	0,536946
2487	0,943904	0,963715	0	0,85025	0,688405	0,980545	0,996569	0,930339	0,930645	0,52073	0,988232	0,997996	0,766833	0,767551	0,536946
2488	0,943904	0,98032	0	0,854417	0,688405	0,980545	0,996569	0,930339	0,930645	0,50052	0,984488	0,997996	0,766833	0,767551	0,533867
2489	0,943904	0,97294	0	0,847917	0,695725	0,980545	0,996569	0,930339	0,930645	0,490357	0,984488	0,997996	0,766833	0,767551	0,536946
2490	0,943904	0,996925	0	0,848667	0,688405	0,980545	0,996569	0,930339	0,930645	0,50052	0,984488	0,997996	0,766833	0,765367	0,533867
2491	0,943904	0,97294	0	0,8495	0,673679	0,980545	0,996569	0,930339	0,930645	0,52073	0,984488	0,996565	0,769327	0,767551	0,53202
2492	0,943904	0,965252	0	0,85025	0,688405	0,980545	0,996569	0,930339	0,930645	0,52073	0,984488	0,99728	0,76995	0,765367	0,535714
2493	0,943904	0,989237	0	0,852833	0,71003	0,980545	0,996569	0,930339	0,930645	0,573738	0,984488	0,995276	0,76995	0,766487	0,535099
2494	0,946522	0,989237	0	0,853667	0,717351	0,980545	0,996569	0,930339	0,930645	0,542326	0,984488	0,997996	0,767768	0,767551	0,538793
2495	0,943904	0,965252	0	0,8575	0,703383	0,980545	0,996569	0,930339	0,930645	0,50052	0,984488	0,998569	0,764651	0,767551	0,538177
2496	0,943904	0,984932	0	0,855167	0,724672	0,980545	0,996569	0,930339	0,930645	0,50052	0,984488	0,996565	0,766833	0,767551	0,54064
2497	0,943904	0,975707	0	0,851833	0,680242	0,980545	0,996569	0,930339	0,930645	0,532279	0,984488	0,996565	0,766833	0,767551	0,538177
2498	0,943904	0,996925	0	0,8495	0,71003	0,980545	0,996569	0,930339	0,930645	0,52073	0,984488	0,995992	0,766833	0,767551	0,536946
2499	0,943904	0,965252	0	0,851833	0,694968	0,980545	0,996569	0,930339	0,930645	0,532279	0,984488	0,997996	0,766833	0,767551	0,536946
2500	0,943904	0,998155	0	0,853667	0,703383	0,980545	0,996569	0,930339	0,930645	0,543481	0,984488	0,997996	0,766833	0,764431	0,534483
2501	0,943904	0,9877	0	0,860167	0,703383	0,980545	0,996569	0,930339	0,929839	0,543481	0,984488	0,997996	0,766833	0,768487	0,529557
2502															
2503															
2504															
2505															
2506															
2507															
2508															
2509															
2510															
2511															
2512															

Gotowy

2. Tabela w programie Excel po sformatowaniu

Widać dokładnie na powyższym zrzucie ekranu, iż wartości danych podawanych na wejścia i wyjście naszej sztucznej sieci neuronowej zawierają się w granicach 0-1. Wnioskiem jest to, iż wszystkie wejścia posiadają obecnie taką samą wagność.

Teraz może nastąpić wklejenie danych do programu Matlab 7.0. W tym celu otwieramy tenże program i w linii poleceń edytujemy nowy skrypt dla poszczególnych danych, które chcemy zapamiętać do wykonania niezbędnych obliczeń.



3. Edycja skryptu

Wciskając klawisz [enter], otrzymujemy dostęp do okna edycji skryptu i tam deklarujemy tablicę wejściową i wyjściową dla treningu i symulacji w następujący sposób: *nazwa zmiennej jest tablicą danych wejściowych po transponowaniu.*

The screenshot shows a MATLAB Editor window titled "Editor - C:\MATLAB7\work\wejtr.m". The script content is as follows:

```

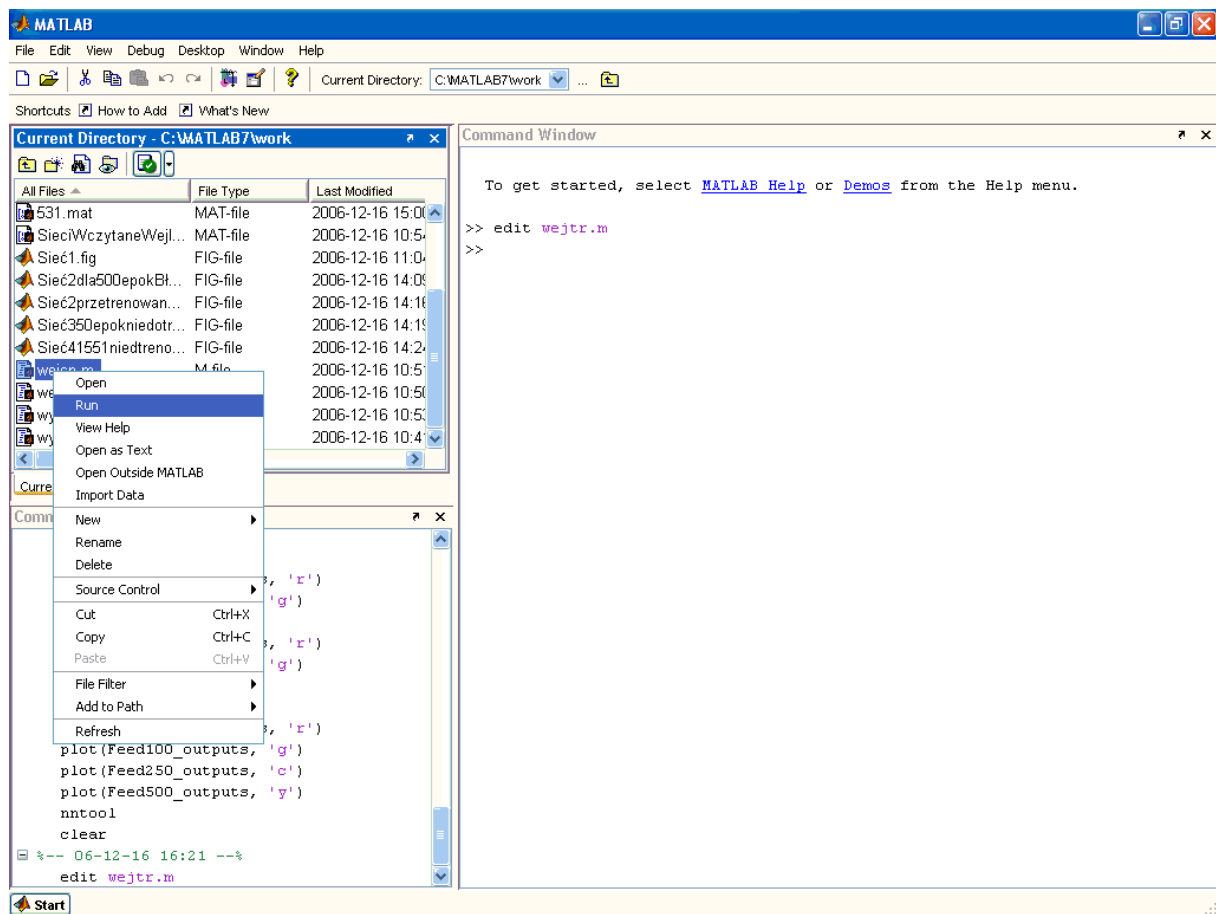
1 - wejtrening= [
2   0.623410621 0.795817958 0 0.847916667 0.739481656 0.983456855 0.997141224 0.995158687 0.995967742 0.520729876 0
3   0.623410621 0.800123001 0 0.846333333 0.702288792 0.983456855 0.997141224 0.995158687 0.995967742 0.50051969 0.987
4   0.623410621 0.800123001 0 0.848666667 0.702288792 0.983456855 0.997141224 0.995158687 0.995967742 0.460099319 0.987
5   0.623410621 0.780750308 0 0.848666667 0.710030293 0.983456855 0.997141224 0.995158687 0.995967742 0.490356854 0.987
6   0.623410621 0.780750308 0 0.85025 0.702288792 0.983456855 0.997141224 0.997310382 0.995967742 0.533317935 0.9870286
7   0.623410621 0.780750308 0 0.85025 0.708936385 0.983456855 0.997141224 0.997310382 0.995967742 0.562651576 0.9870286
8   0.623410621 0.792742927 0 0.85025 0.710030293 0.983456855 0.997141224 0.997310382 0.995967742 0.562651576 0.9870286
9   0.626028422 0.792742927 0 0.845333333 0.694968024 0.983456855 0.996569468 0.995158687 0.995967742 0.542325904 0.987
10  0.626028422 0.771525215 0 0.847916667 0.732076742 0.983456855 0.996569468 0.995158687 0.99811828 0.583901143 0.987
11  0.626028422 0.802583026 0 0.851 0.739481656 0.983456855 0.996569468 0.997310382 0.995967742 0.583901143 0.9870286
12  0.623410621 0.801353014 0 0.846333333 0.724671828 0.983456855 0.996569468 0.997310382 0.995967742 0.583901143 0.987
13  0.623410621 0.802583026 0 0.8495 0.762537866 0.983456855 0.996569468 0.995158687 0.995967742 0.542325904 0.9870286
14  0.623410621 0.802583026 0 0.8495 0.710030293 0.983456855 0.996569468 0.995158687 0.995967742 0.572698926 0.9870286
15  0.623410621 0.785055351 0 0.851 0.710030293 0.983456855 0.996569468 0.995158687 0.995967742 0.583901143 0.9870286
16  0.623410621 0.785055351 0 0.847166667 0.71735106 0.983456855 0.997141224 0.997310382 0.995967742 0.583901143 0.987
17  0.623410621 0.800123001 0 0.848666667 0.724671828 0.983456855 0.997141224 0.997310382 0.99811828 0.59406398 0.987
18  0.623410621 0.819803198 0 0.856 0.710030293 0.979354156 0.997141224 0.995158687 0.995967742 0.55248874 0.9870286
19  0.623410621 0.81795818 0 0.852833333 0.710703467 0.979354156 0.997141224 0.995158687 0.995967742 0.573738307 0.987
20  0.623410621 0.81795818 0.994949495 0.84375 0.732834063 0.979354156 0.997141224 0.995158687 0.995967742 0.59406398 0
21  0.623410621 0.828720787 0.988344988 0.847166667 0.725513295 0.979354156 0.997141224 0.995158687 0.996774194 0.5423259
22  0.623410621 0.813345633 0.984848485 0.847166667 0.703382699 0.979354156 0.997141224 0.995158687 0.996774194 0.5636909
23  0.623410621 0.834563346 0.954545455 0.84375 0.703382699 0.979354156 0.997141224 0.997310382 0.996774194 0.573738307 0
24  0.623410621 0.834563346 0.937062937 0.840666667 0.718108381 0.979354156 0.996569468 0.995158687 0.996774194 0.5423259
25  0.623410621 0.804735547 0.93006993 0.842166667 0.710703467 0.979354156 0.996569468 0.995158687 0.996774194 0.5423259
26  0.623410621 0.800123001 0.926961927 0.8445 0.666273982 0.979354156 0.996569468 0.995158687 0.996774194 0.59406398 0
27  0.623410621 0.811808118 0.925796426 0.846333333 0.681083726 0.979354156 0.996569468 0.995158687 0.99811828 0.5839011
28  0.626776365 0.819803198 0.925019425 0.847166667 0.710703467 0.979354156 0.996569468 0.995158687 0.99811828 0.5536436
29  0.626776365 0.807503075 0.926184926 0.841333333 0.710703467 0.979354156 0.996569468 0.995158687 0.997311828 0.5207298
30  0.629020194 0.831488315 0.926961927 0.840666667 0.710703467 0.979354156 0.996569468 0.995158687 0.997311828 0.5322785
31  0.629020194 0.82103321 0.927544678 0.8445 0.694968024 0.979354156 0.996569468 0.995158687 0.997311828 0.564730338 0
32  0.629020194 0.801353014 0.928515929 0.841333333 0.688404578 0.979354156 0.996569468 0.995158687 0.997311828 0.5423259
33  0.629020194 0.810578106 0.929487179 0.838 0.732834063 0.979354156 0.996569468 0.995158687 0.997311828 0.542325904 0
34  0.629020194 0.80904059 0.930846931 0.840666667 0.703382699 0.979354156 0.997141224 0.997310382 0.997311828 0.5333179
35  0.629020194 0.834563346 0.930846931 0.840666667 0.703382699 0.979354156 0.997141224 0.997310382 0.997311828 0.5333179
36  0.629020194 0.800123001 0.93006993 0.84375 0.725513295 0.979354156 0.997141224 0.997310382 0.997311828 0.543480771 0

```

4. Wprowadzenie całej tablicy do skryptu

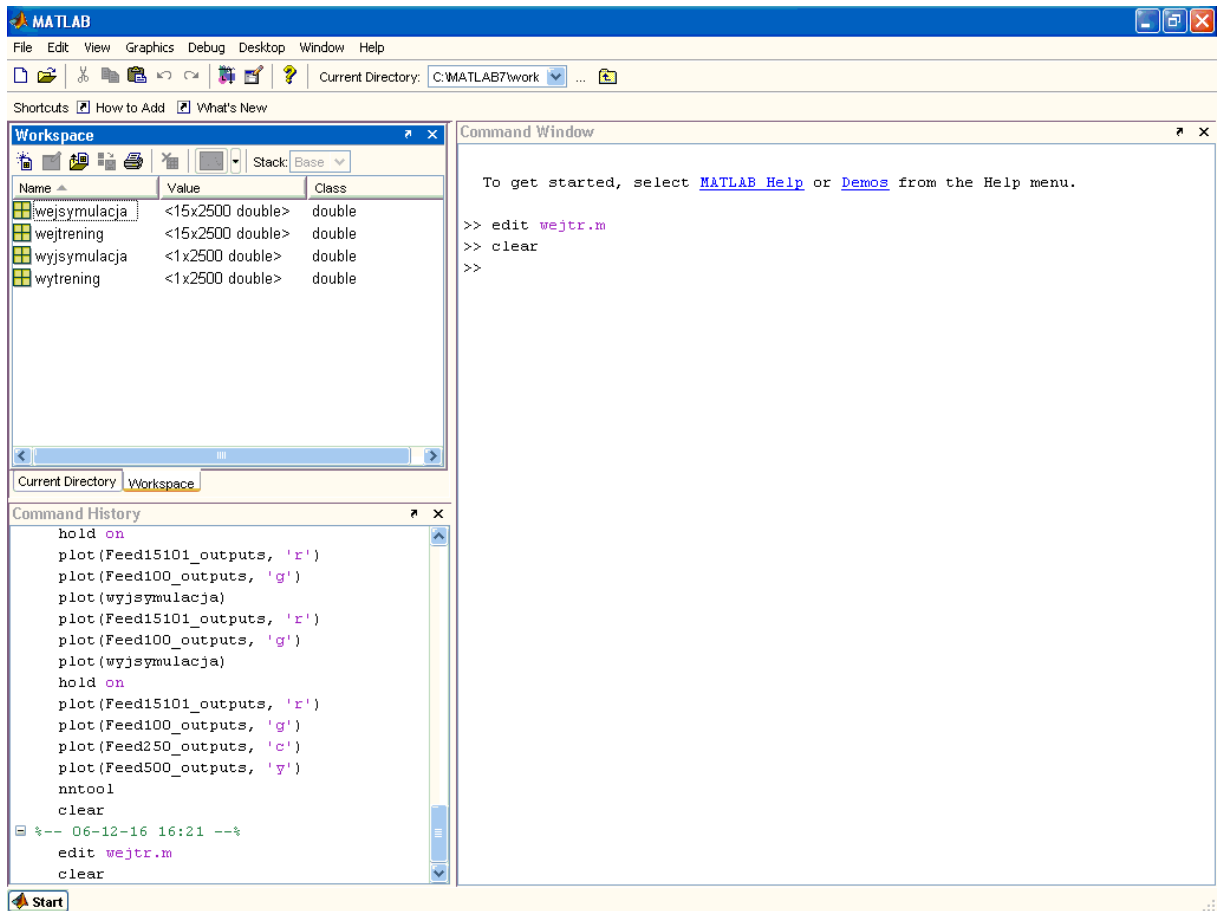
Należy pamiętać przy tym, iż zapis liczb ułamkowych jest różny dla każdego z programów. W programie Matlab jest to kropka dziesiętna, a w Excel jest to przecinek. Dlatego dokonujemy konwersji tych znaków na prawidłowe.

Operację powtarzamy kolejno dla wejść i wyjść sieci, uzyskując w ten sposób cztery tablice. Dalej, wczytujemy zmienne, uruchamiając skrypty po kolei.



5. Wczytanie tablicy poprzez wybranie opcji RUN z listy rozwijanej

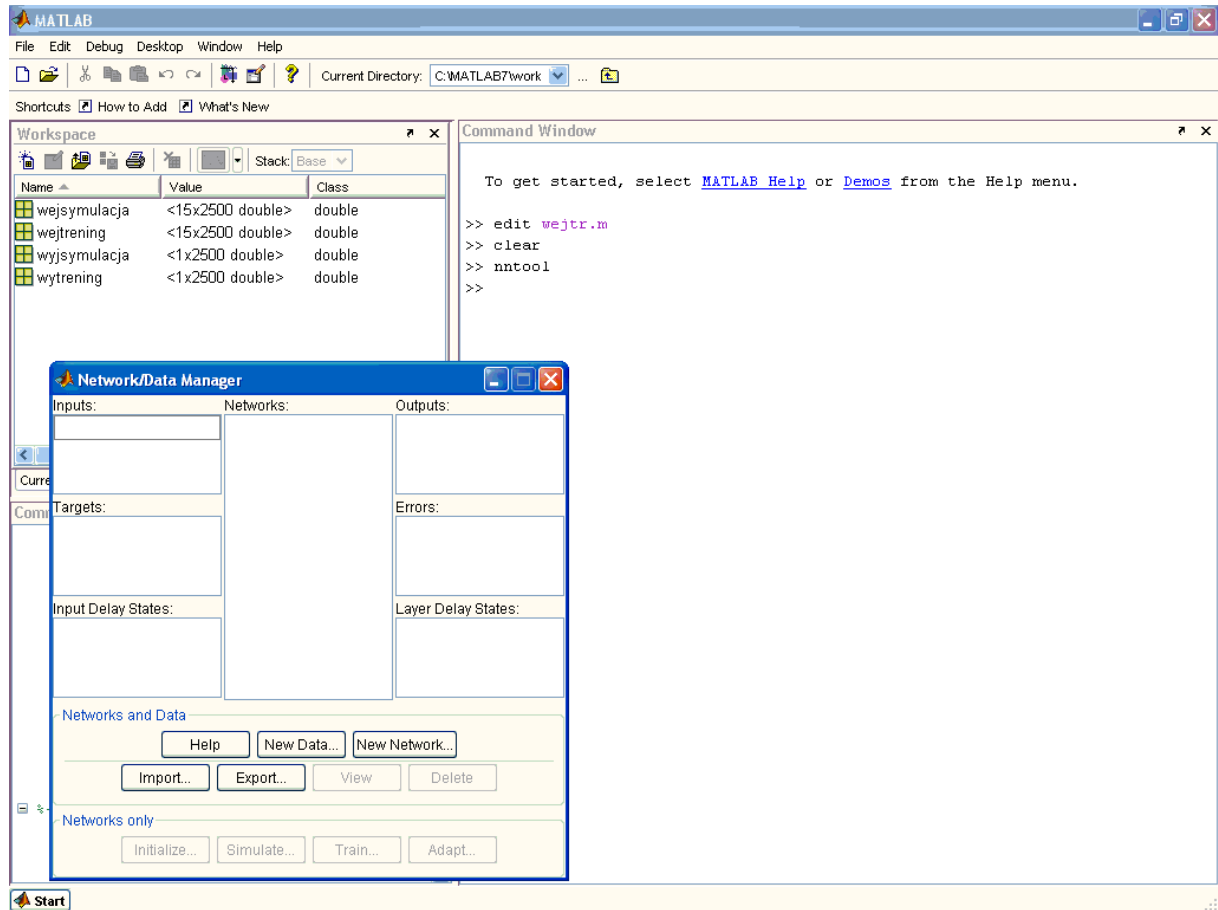
Otrzymujemy w ten sposób dostęp do tablic.



6. Tablice widoczne w lewym górnym oknie

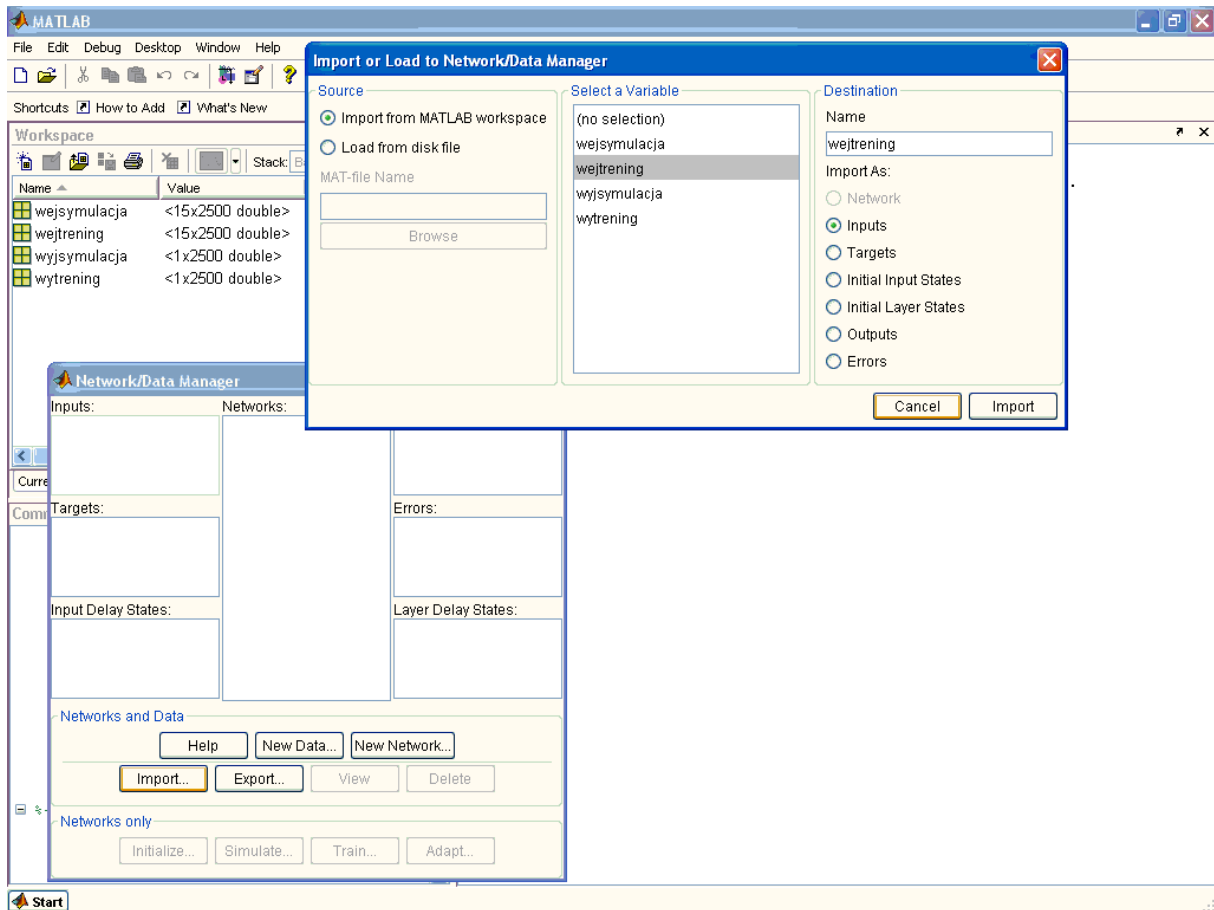
1.2 Inicjalizacja narzędzia Neural Networks Tool oraz procedura wczytania danych, treningu sieci oraz symulacji

Całą procedurę rozpoczynamy od wpisania komendy nntool w linii poleceń programu Matlab 7.0. Skutkować to będzie pojawieniem się okna przedstawionego poniżej.



7. Okno narzędzia NNTOOL

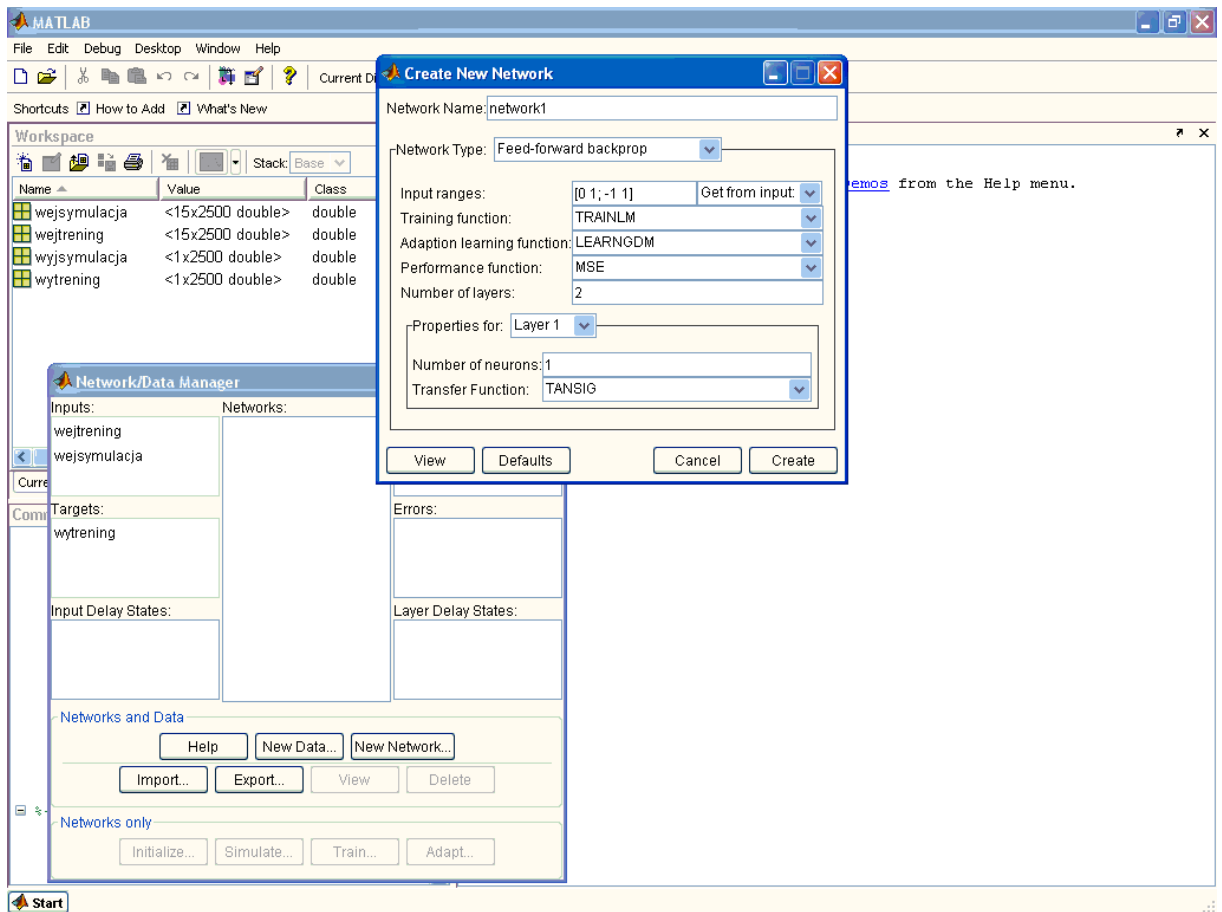
Teraz wczytujemy wejścia i wyjścia naszej sieci używając przycisku *Import*. Pojawi się okno przedstawione poniżej, gdzie określamy, czy dane przez nas wybrane mają być przypisane do wejścia, czy wyjścia sieci (Inputs/Targets).



8. Wybór tablicy wejtrening jako wejściowa

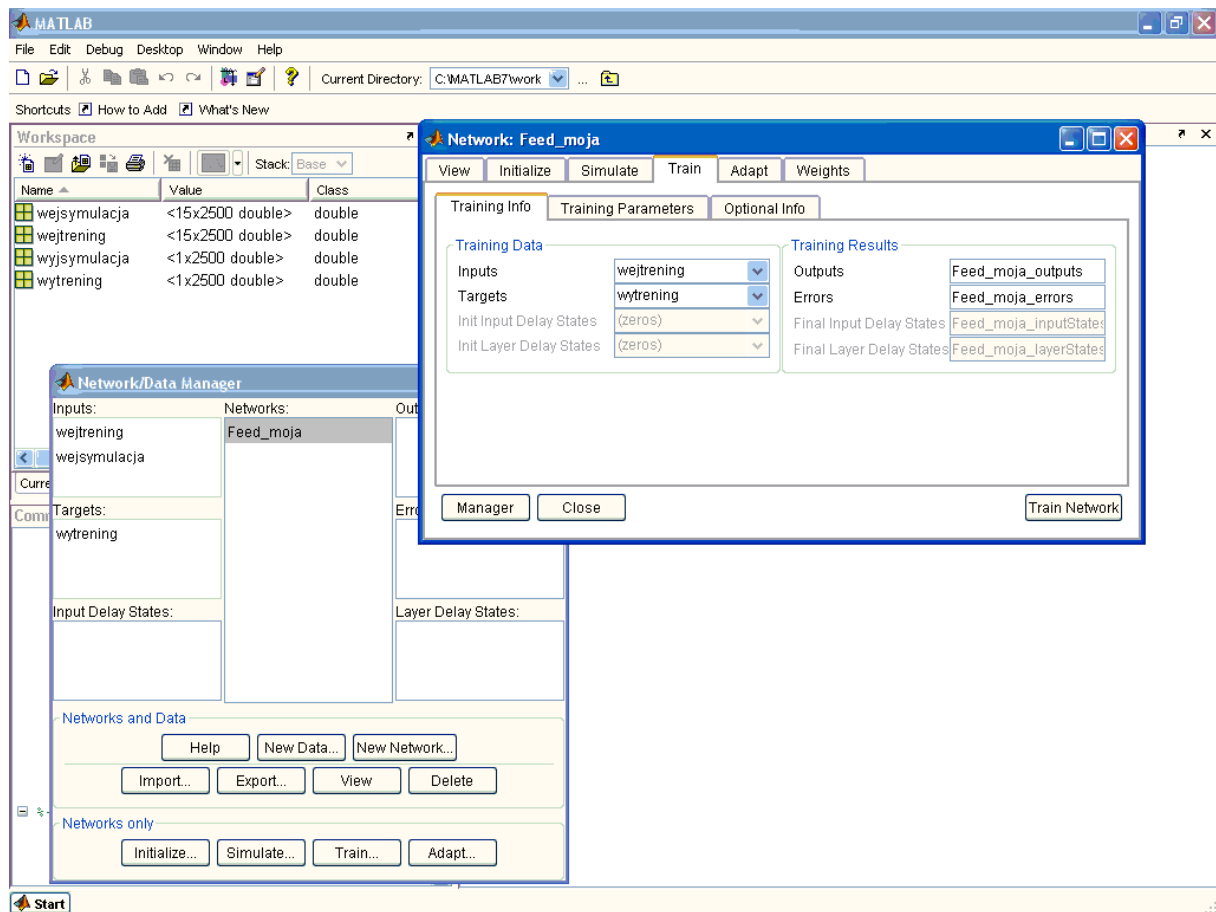
Po zadeklarowaniu naszych danych, naciskamy przycisk *New Network...* celem zaprojektowania naszej sieci neuronowej. Z ważniejszych, po kolei możemy zmienić:

- ⇒ Nazwę
- ⇒ Rodzaj sieci
- ⇒ Zakres danych wejściowych
- ⇒ Liczbę warstw
- ⇒ Liczbę neuronów w poszczególnej warstwie
- ⇒ Funkcję przejścia



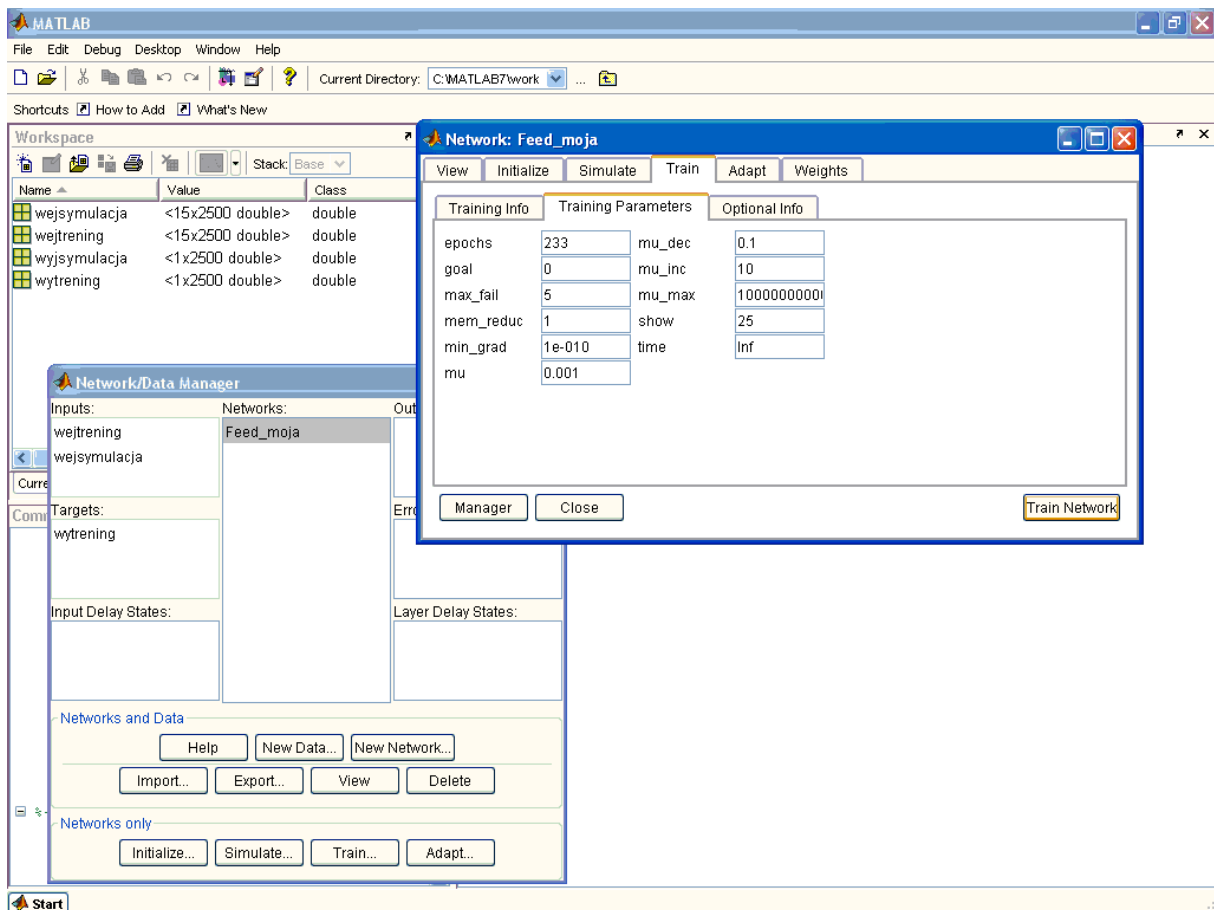
9. Deklarowanie wszelkich parametrów naszej sieci neuronowej

Po skonfigurowaniu naszej sieci następuje faza treningu. Rozpoczynamy ją naciskając przycisk *Train...* z dostępnych tym oknie. Pojawi się okno pokazane poniżej.



10. Okno treningu z otwartą zakładką *Training Info*

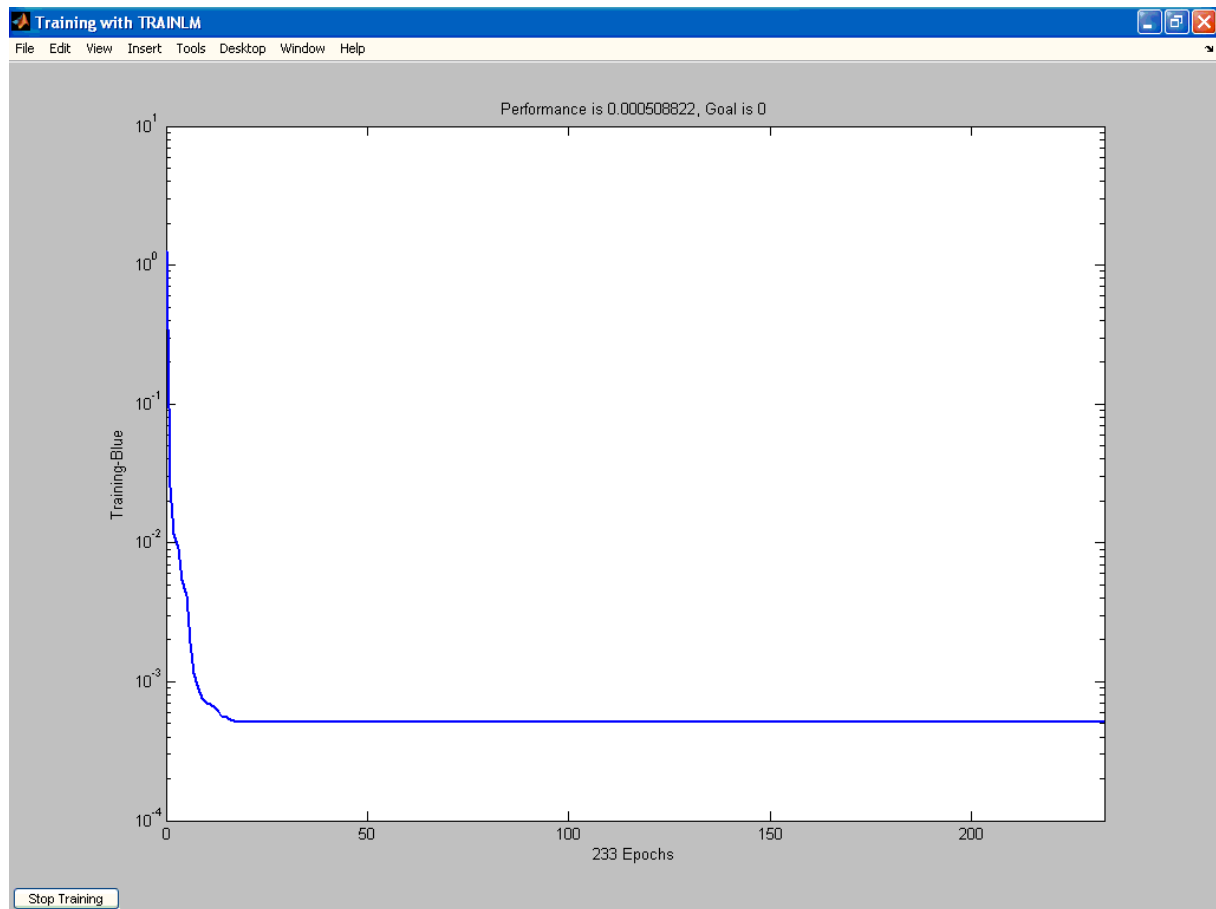
Deklarujemy tutaj, które zmienne mają być danymi podawanymi na wejścia sieci neuronowej które mają być informacją przedstawianą sieci na wyjściu oraz mamy możliwość zmiany nazw zmiennych z wynikami treningu tzn. tablicy z wyjściami oraz tablicy z błędami. Po skonfigurowaniu wszelkich interesujących nas informacji w tej zakładce, przechodzimy do kolejnej, czyli parametrów treningu. Tu wybieramy liczbę epok (u mnie 233) oraz do jakiej wartości dążyć ma błąd lub, kiedy przerwać trening sieci (wyniki satysfakcjonujące). Pokazane jest to poniżej na zrzucie ekranu.



11. Wybór liczby epok uczenia sieci

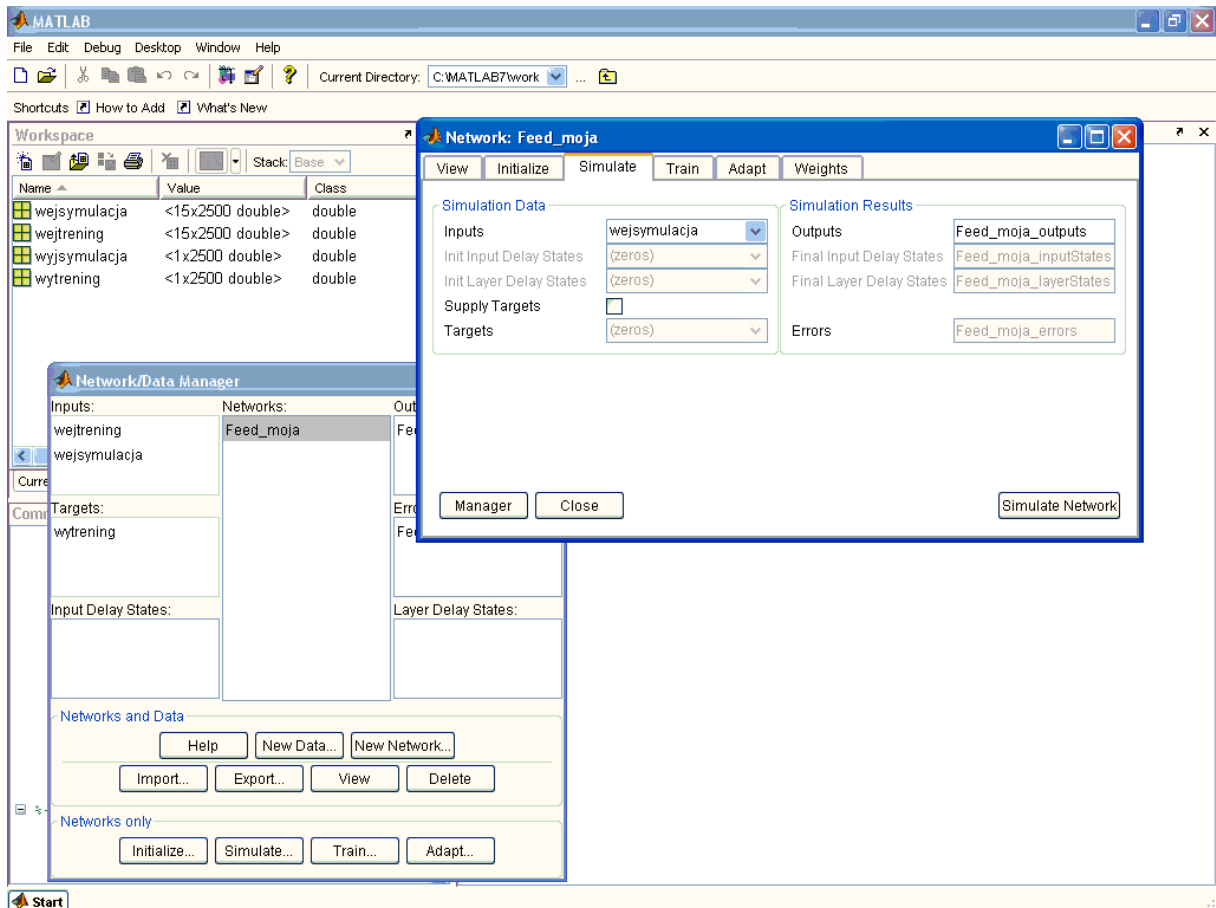
Kliknięcie na przycisku *Train Network* powoduje rozpoczęcie fazy treningu sieci, co zobrazowane jest na dynamicznie zmieniającym się wykresie, dążącym do wykonania zadeklarowanych u nas dwustu trzydziestu trzech epok. Na zrzucie ekranu

przedstawionym poniżej, doskonale widać, jak zmienia się błąd wraz z narastaniem epok.



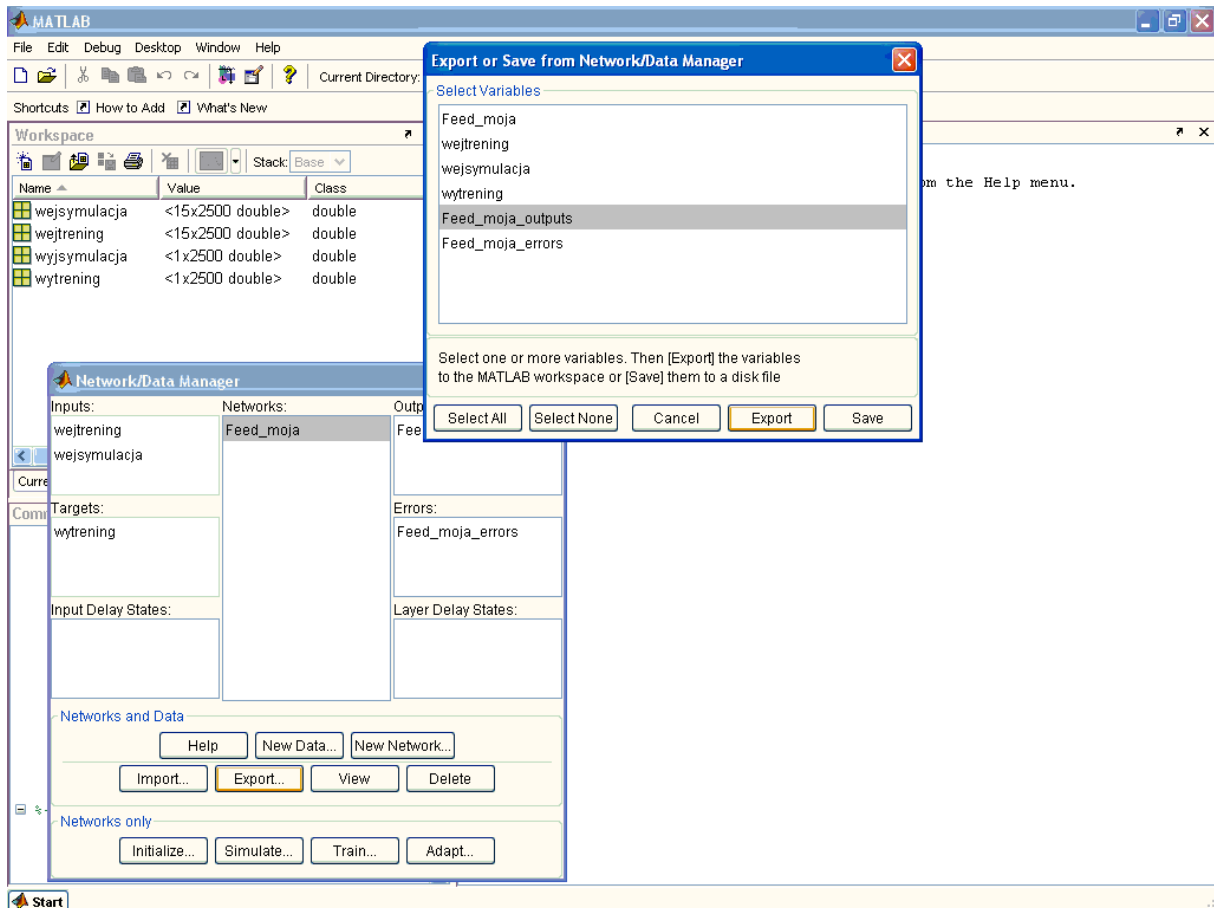
12. Błąd generowany przez naszą sieć po 233 epokach

Okno to zamykamy i wybieramy zakładkę *Simulate*, gdzie wybieramy nasze dane sprawdzające, które mają zostać podane na wejście sieci neuronowej celem weryfikacji jej sprawności uczenia. Na zrzucie ekranu pokazanym niżej widać wybrane dane. Naciśnięcie przycisku *Simulate Network* rozpocznie procedurę przedstawiania sieci próbek sprawdzających.



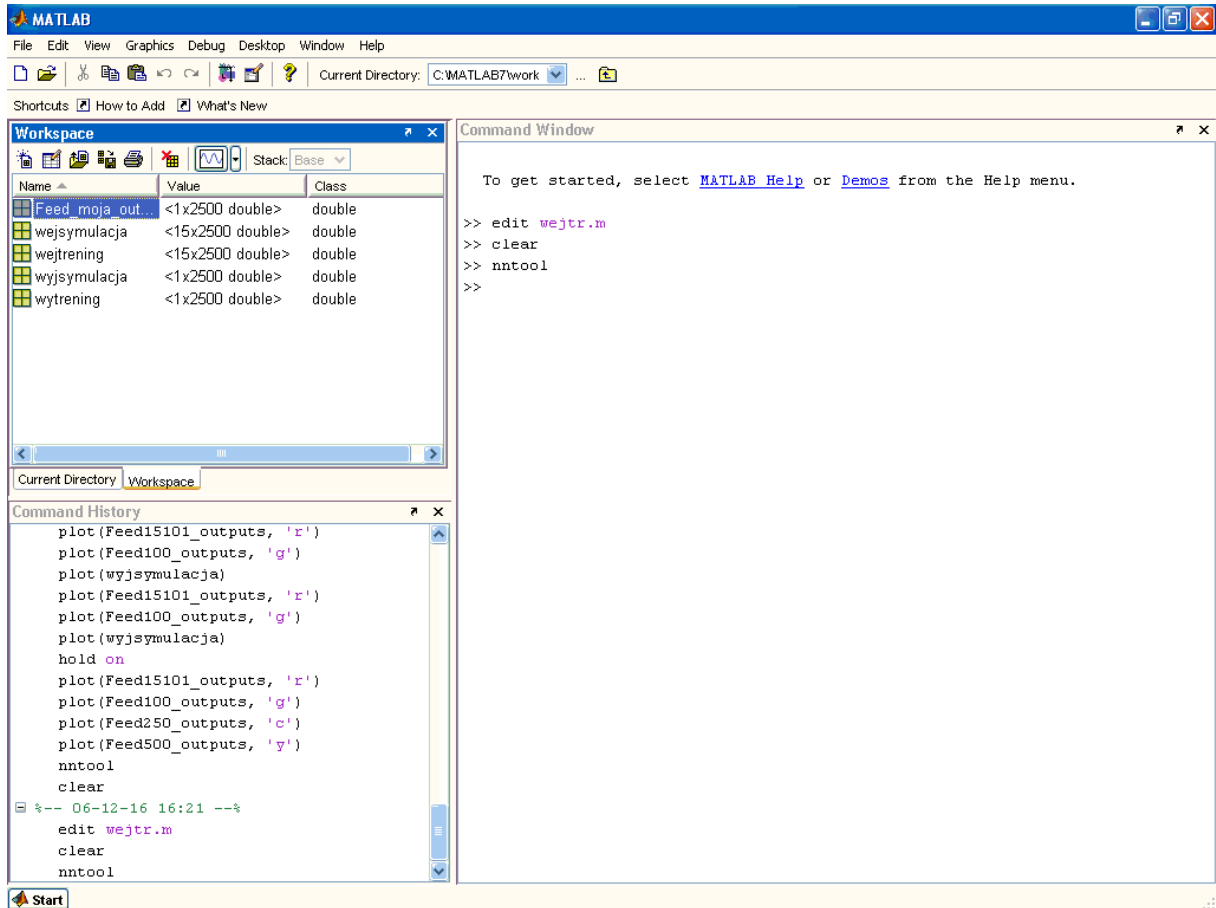
13. Sprawdzenie sieci

Aby móc wizualizować wyniki naszej pracy, musimy eksportować nasze dane do okna głównego programu Matlab 7.0. W tym celu klikamy na przycisku *Export* i wybieramy interesujące nas dane (w moim przypadku jest to tablica z danymi otrzymanymi na wyjściu sieci. Patrz zrzut ekranu niżej.



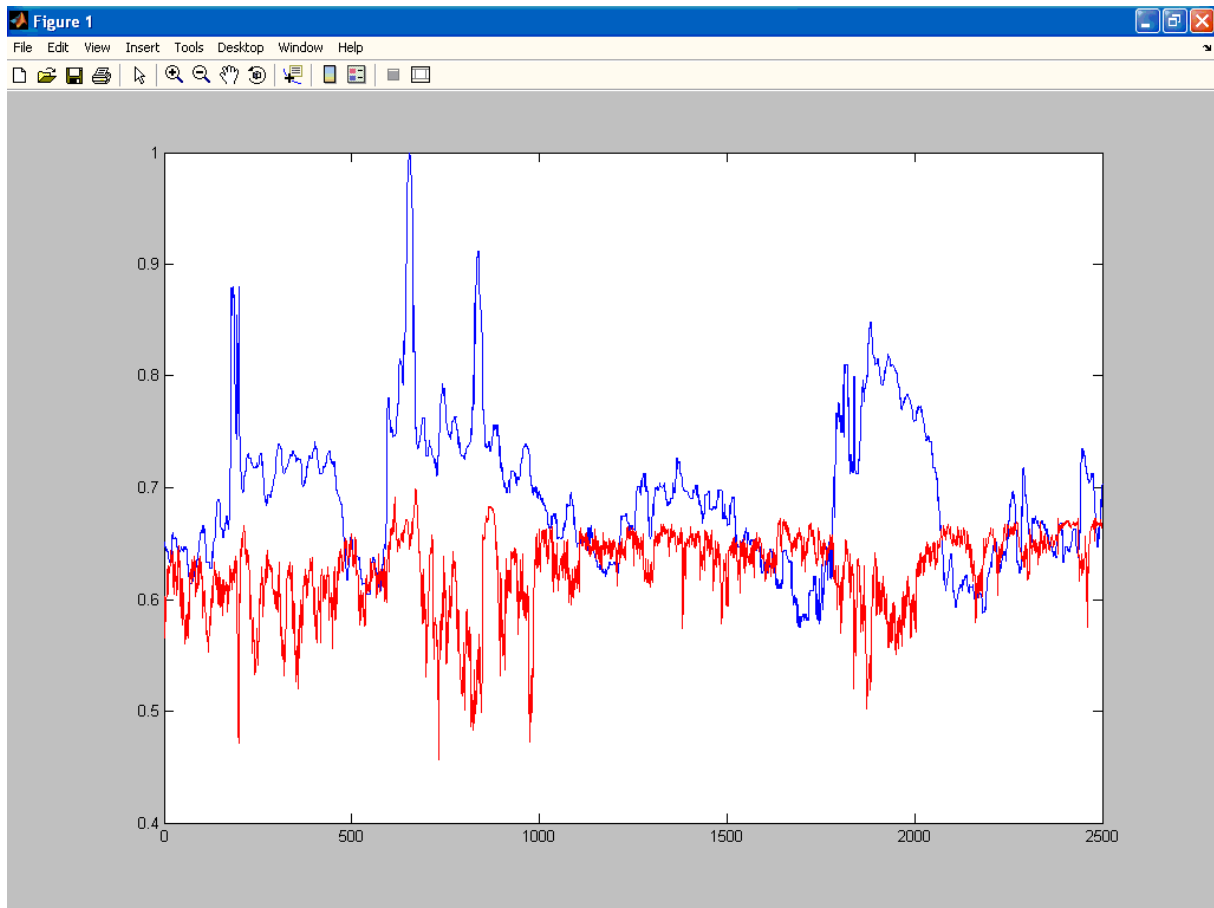
14. Eksportowanie zmiennej wyjściowej

Po przełączeniu się do okna programu głównego, widzimy zmienną pożądaną przez nas, w lewym górnym oknie programu. Jak widać, na zrzucie ekranu poniżej, ma ta tablica wymiary 1x2500, czyli wymiar się zgadza z liczbą próbek przedstawianych wyjściu sieci w fazie treningu.



15. Okno główne i interesujące nas dane wynikowe

Ostatnią fazą jest przedstawienie porównania rzeczywistych próbek wyjściowych obiektu oraz próbek uzyskanych w procesie symulacji. Na poniższym zrzucie ekranu pokazuję już ostateczny wynik procesu uczenia.



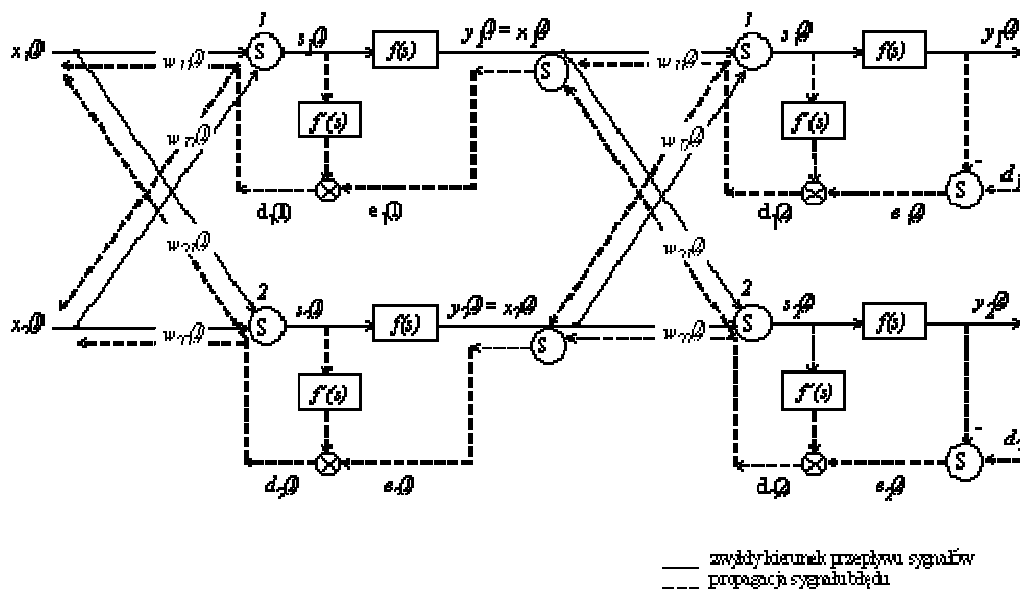
16. Wyniki uczenia

Niebieski wykres, to próbki z rzeczywistego obiektu, natomiast czerwony, to sposób, w jaki sieć poradziła sobie z procesem uczenia. Dalej można stosować proces obróbki statystycznej danych pod względem błędów, czego tu nie będę przedstawiał.

2 Wyniki otrzymanych symulacji poszczególnych sieci neuronowych

2.1 Sieci Feed-forward backpropagation

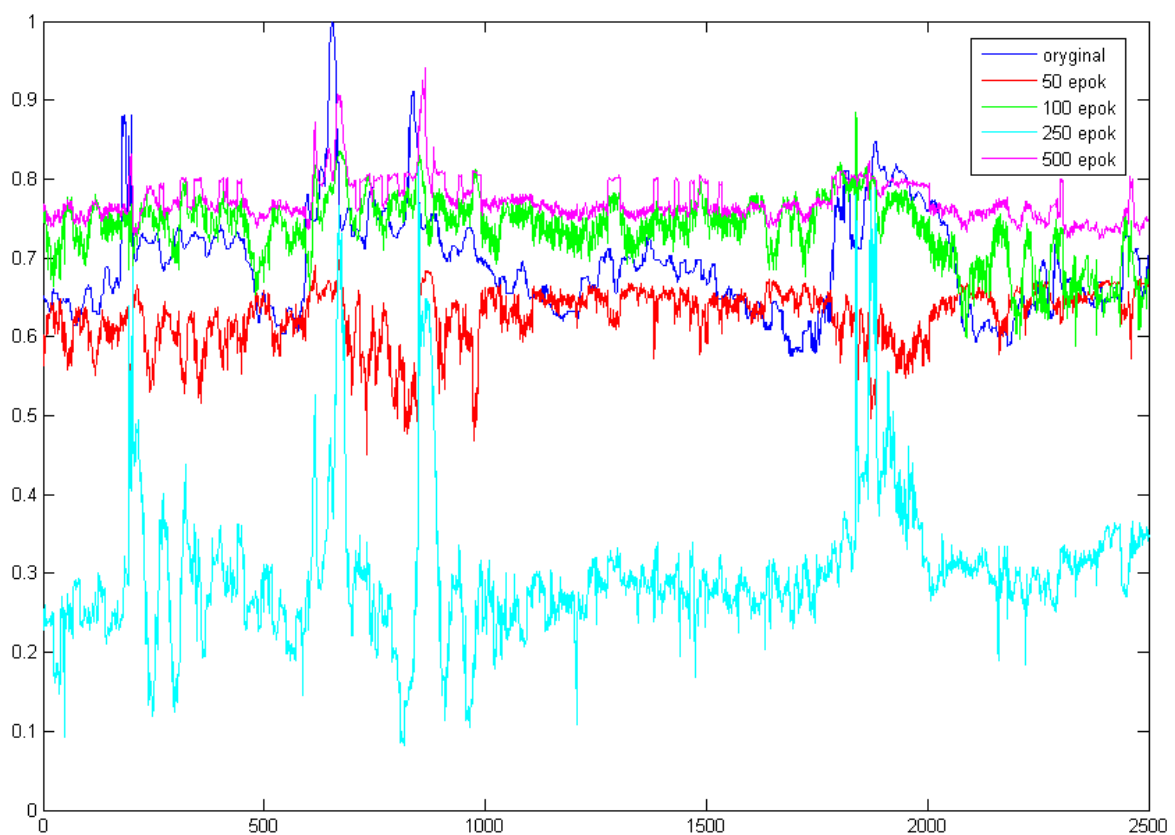
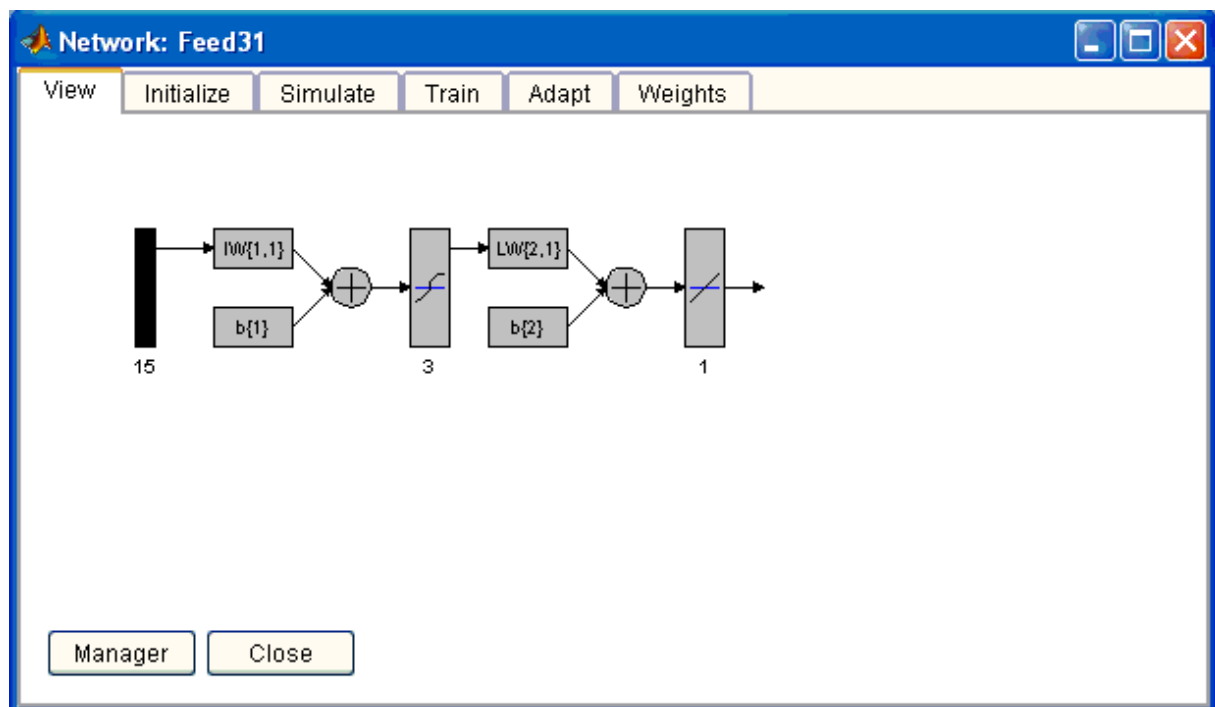
Krótki opis sieci:



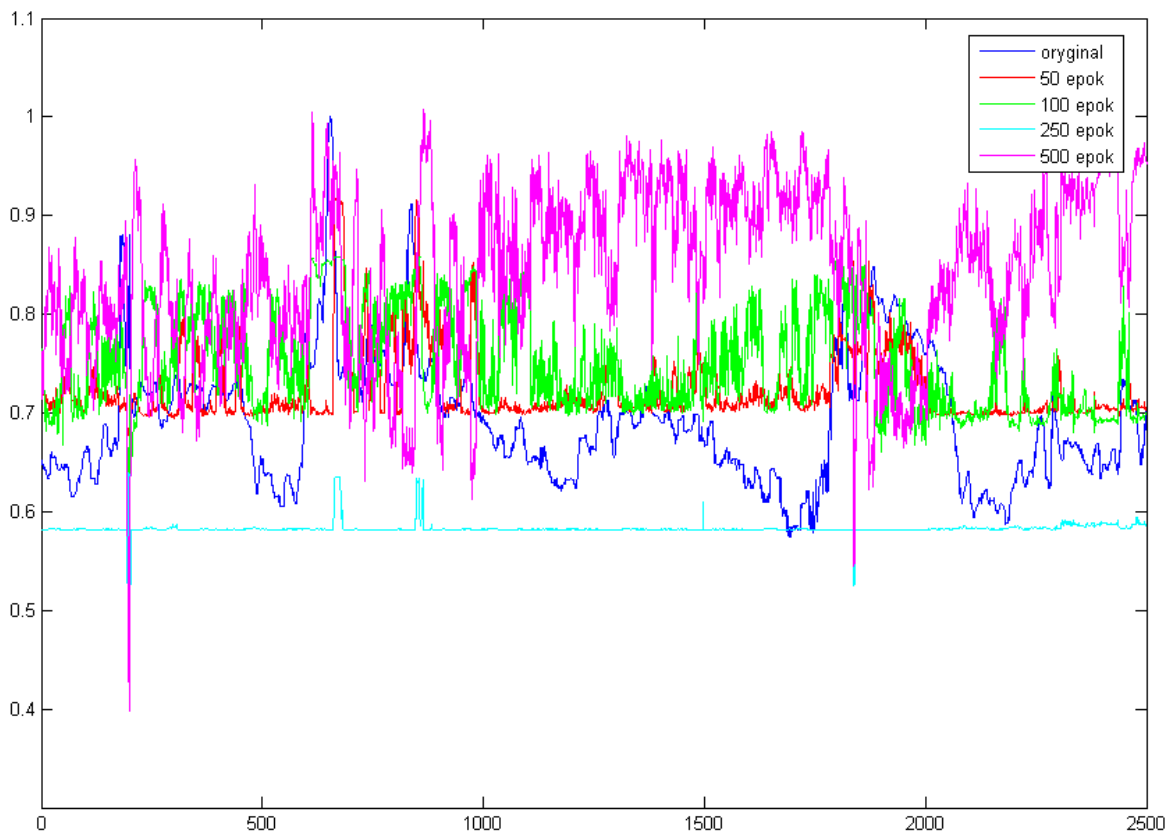
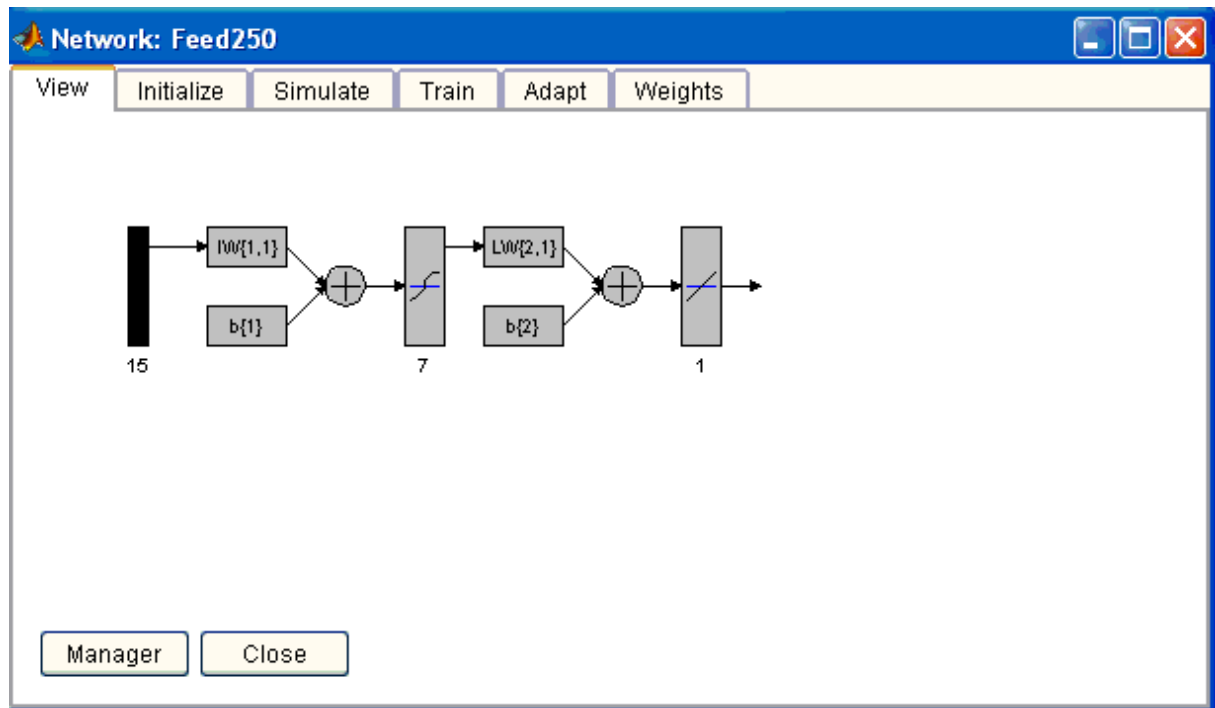
17. Przykładowa sieć jednokierunkowa feed-forward

- ✓ Jest to sieć jednokierunkowa
- ✓ Występuje tutaj propagacja wsteczna błędów
- ✓ Jedna z najczęściej wykorzystywanych sieci neuronowych
- ✓ Inaczej zwana perceptronem wielowarstwowym
- ✓ Zwykle uczenie odbywa się z nauczycielem, czyli jest nadzorowane. Pokazujemy próbki na wejściu sieci i jednocześnie wyniki, jakie chcemy uzyskać

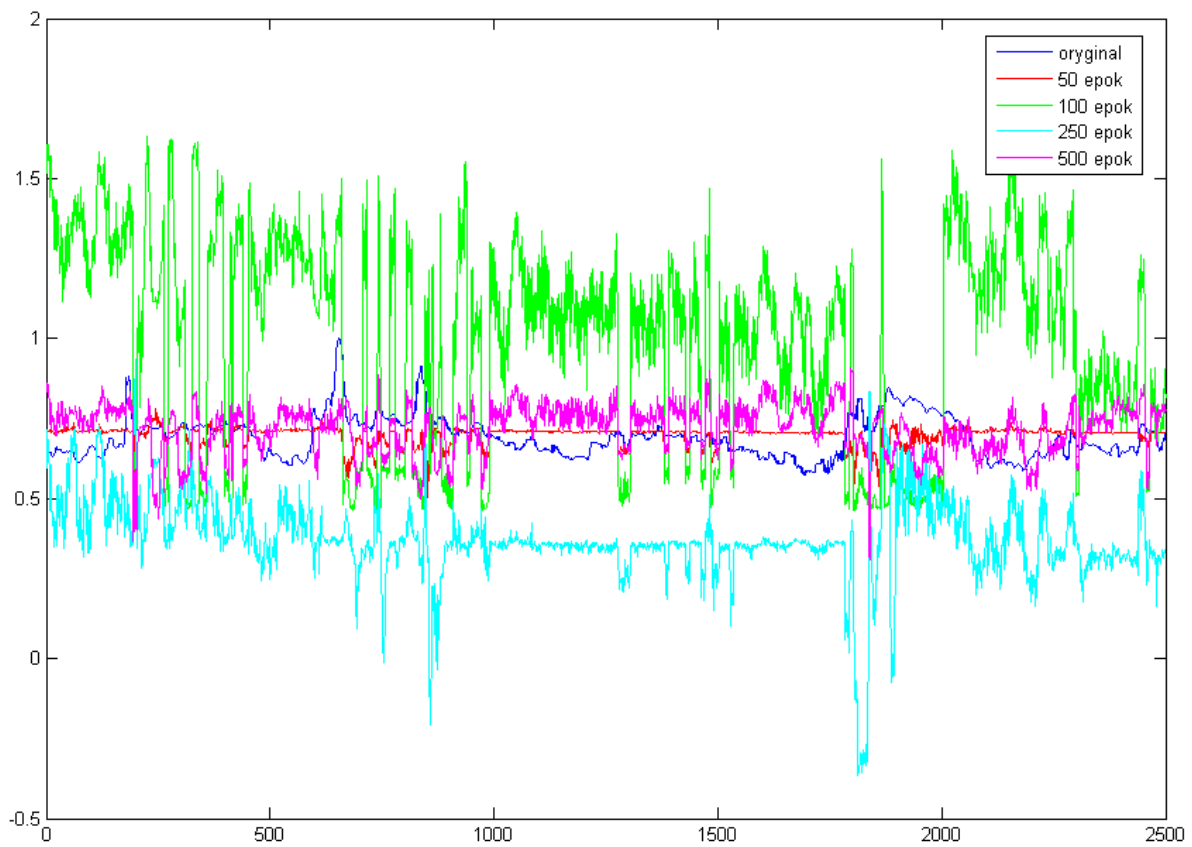
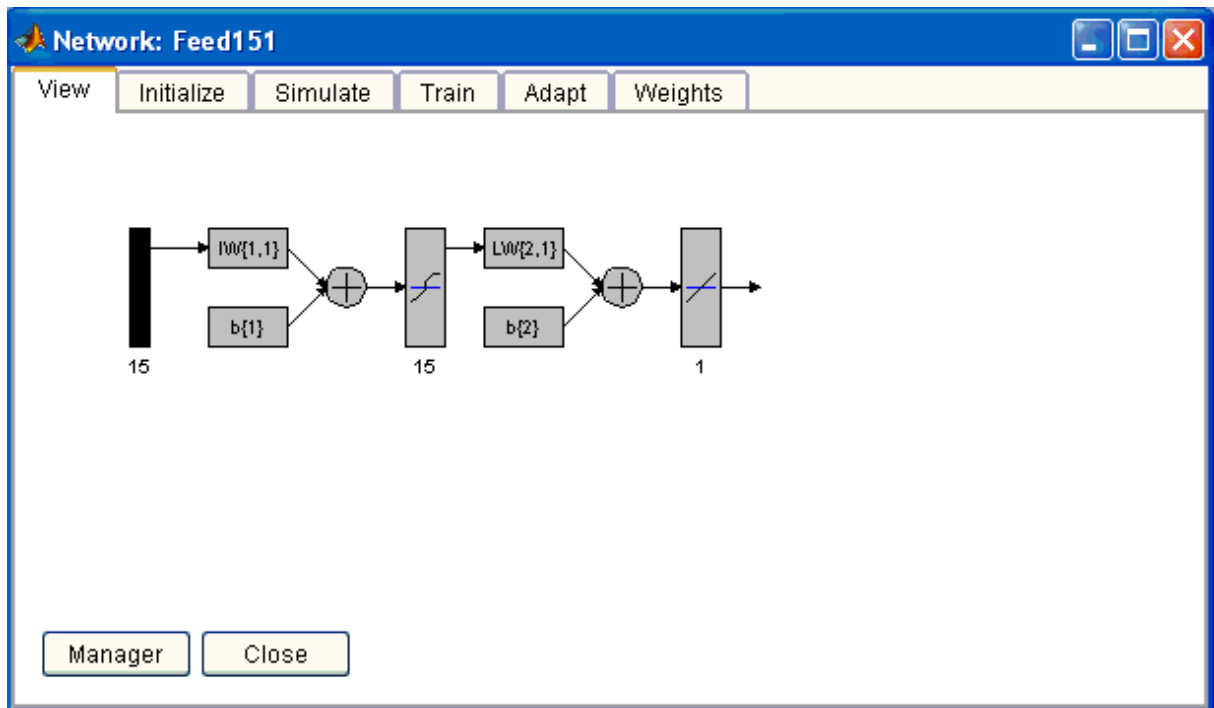
2.1.1 Dwie warstwy ukryte z ilością neuronów 3-1



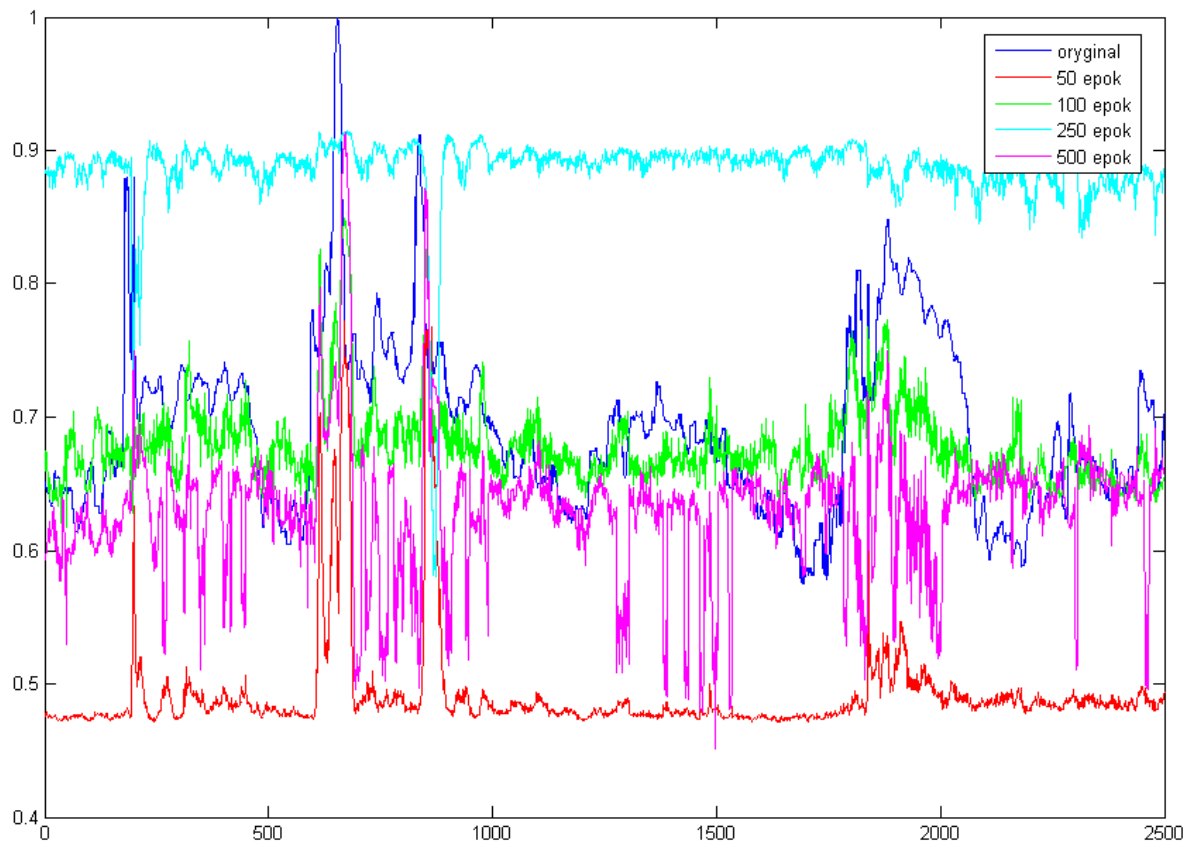
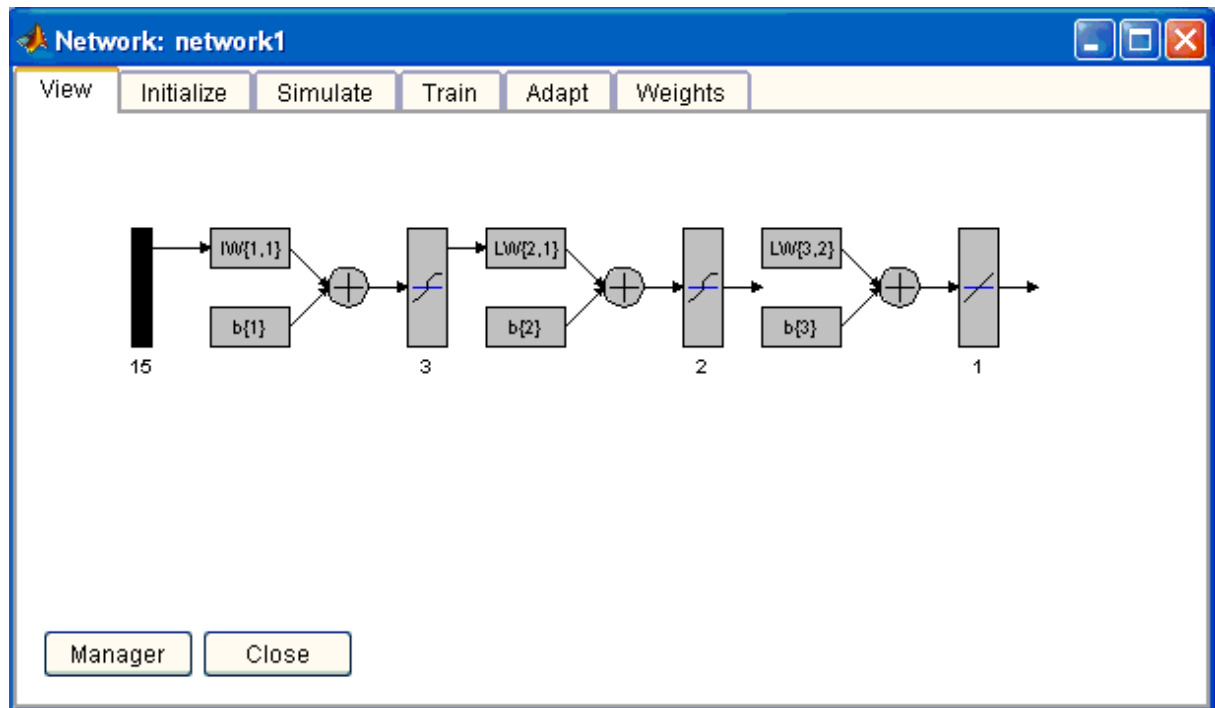
2.1.2 Dwie warstwy ukryte z ilością neuronów 7-1



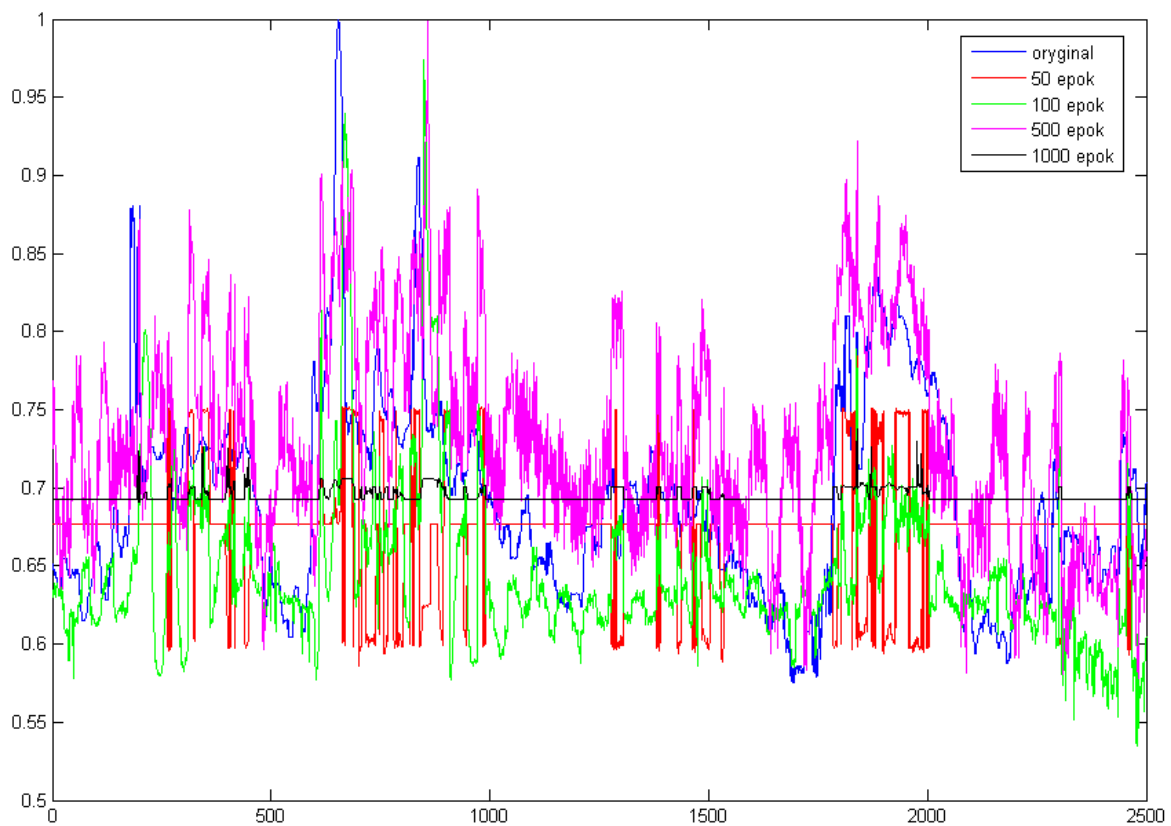
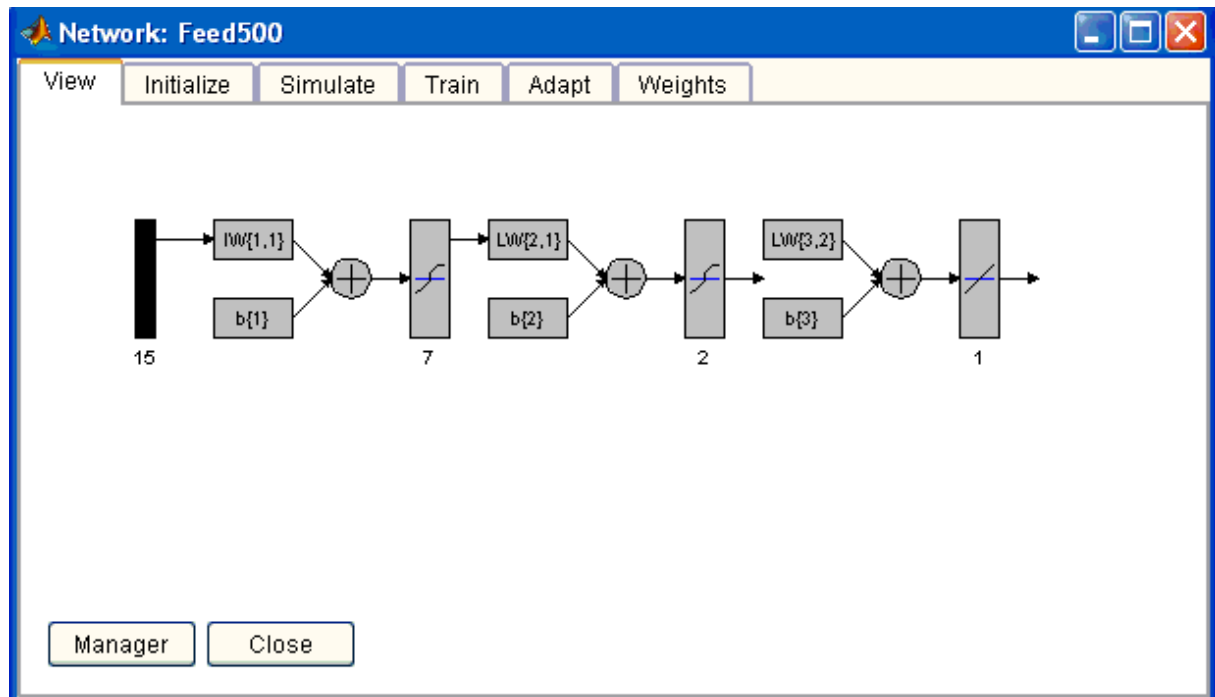
2.1.3 Dwie warstwy ukryte z ilością neuronów 15-1



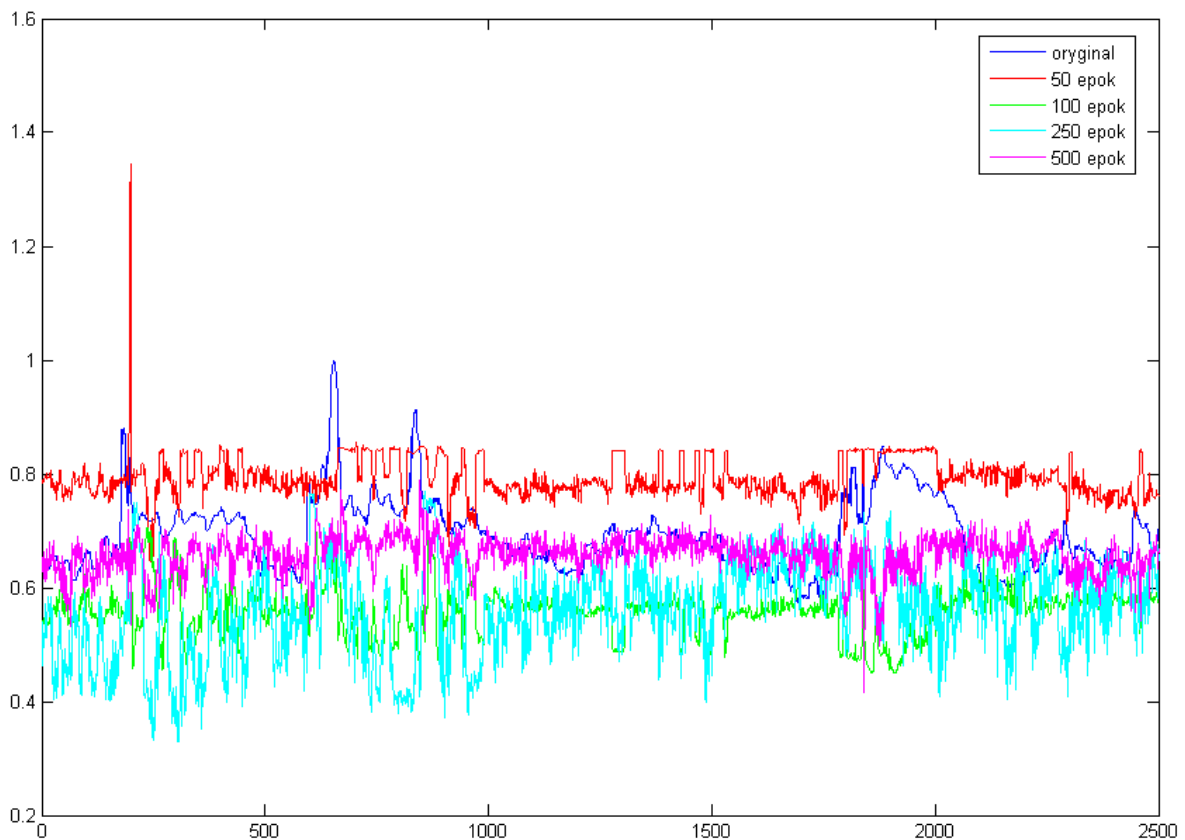
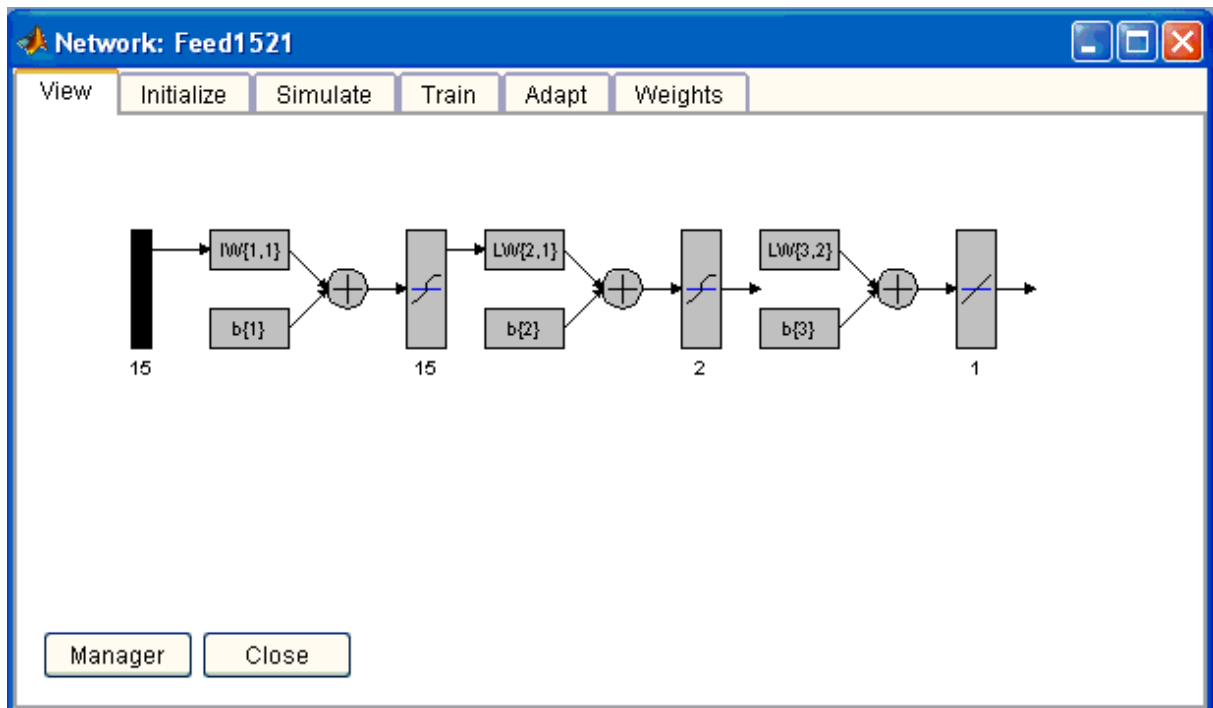
2.1.4 Trzy warstwy ukryte z ilością neuronów 3-2-1



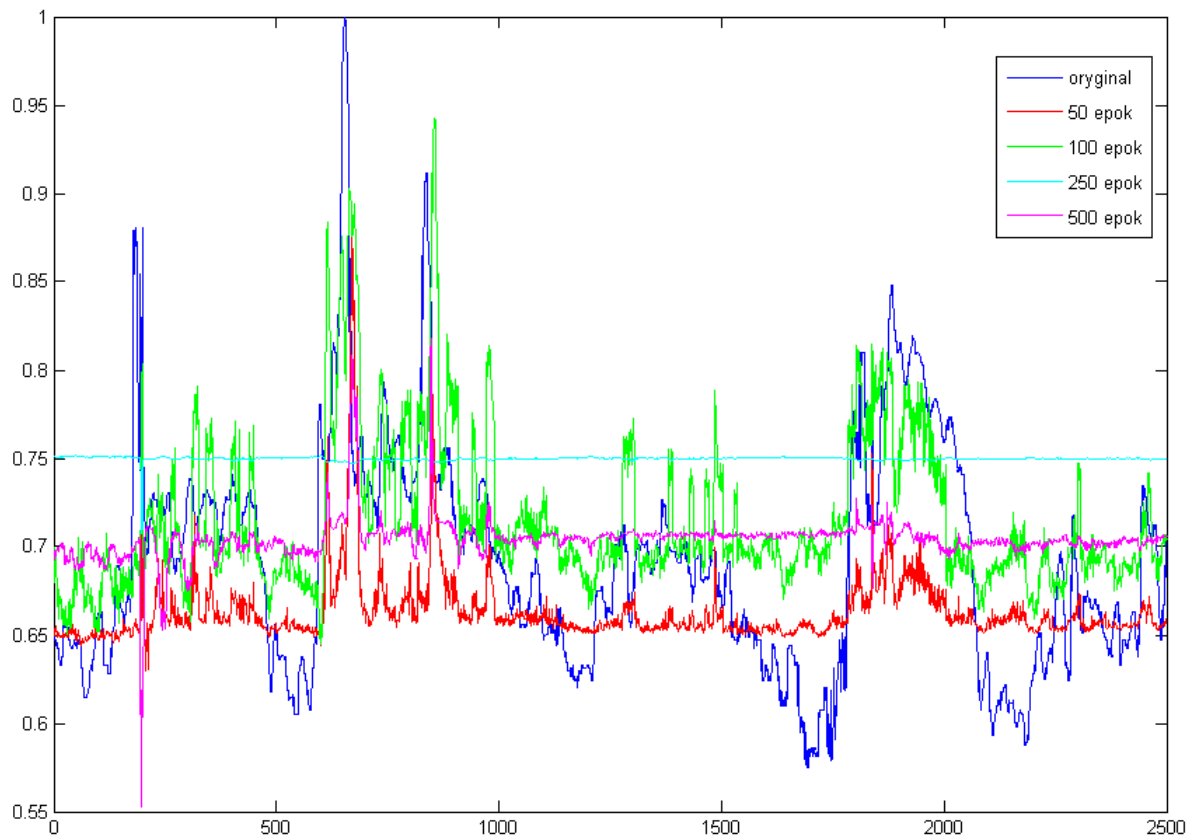
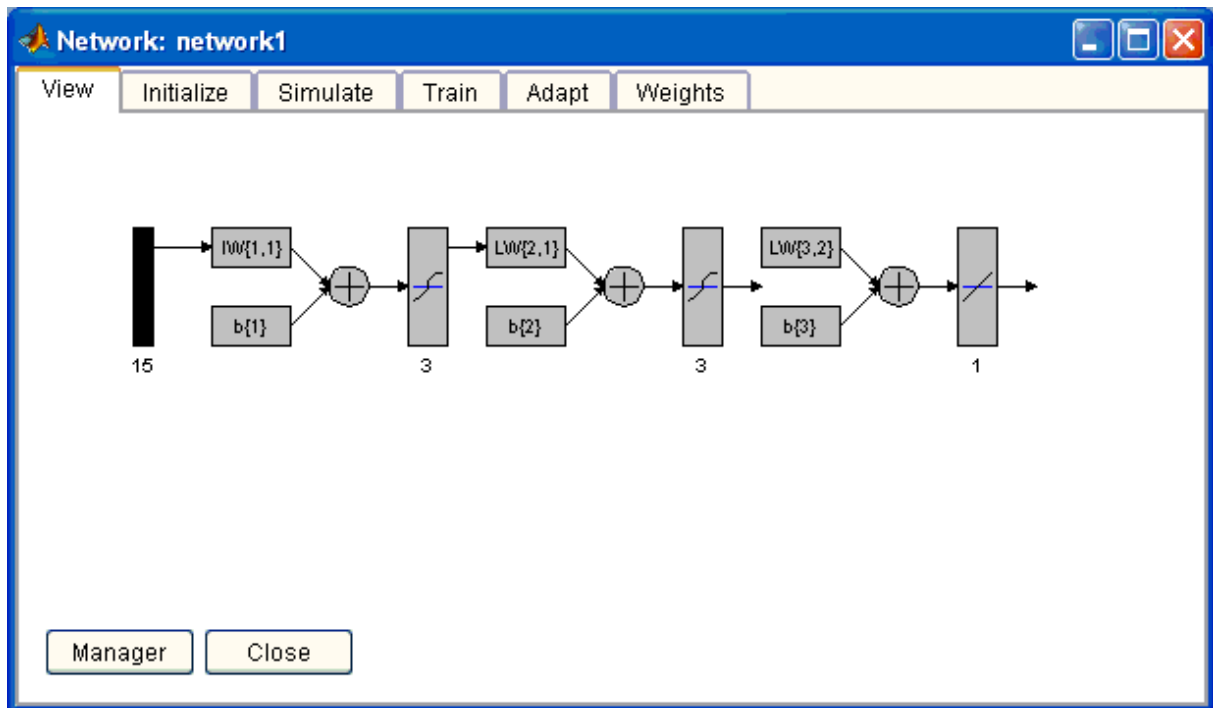
2.1.5 Trzy warstwy ukryte z ilością neuronów 7-2-1



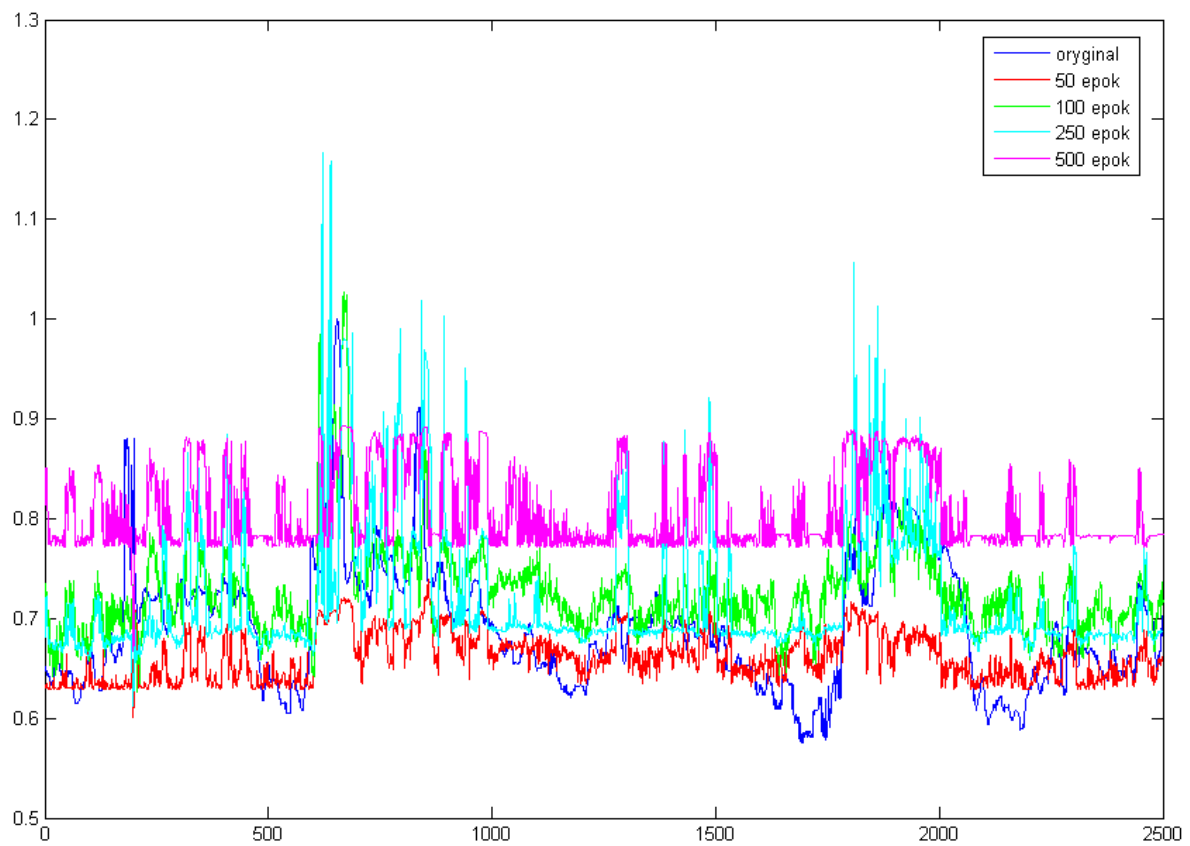
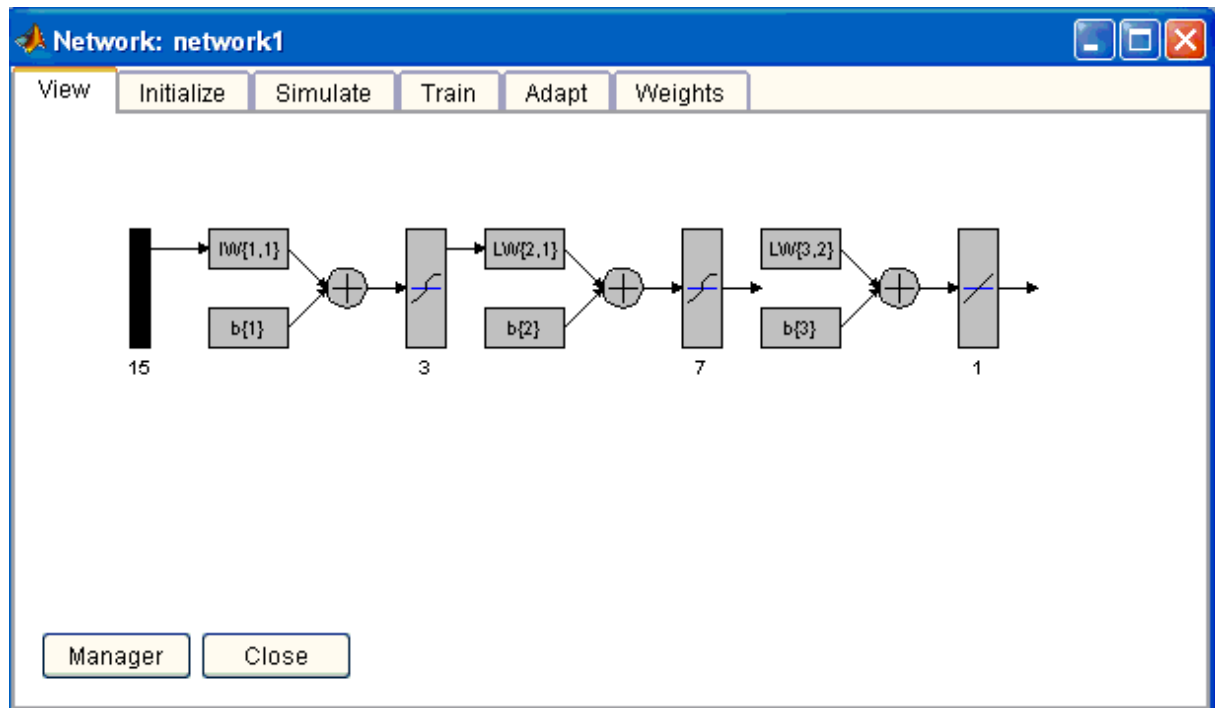
2.1.6 Trzy warstwy ukryte z liczbą neuronów 15-2-1



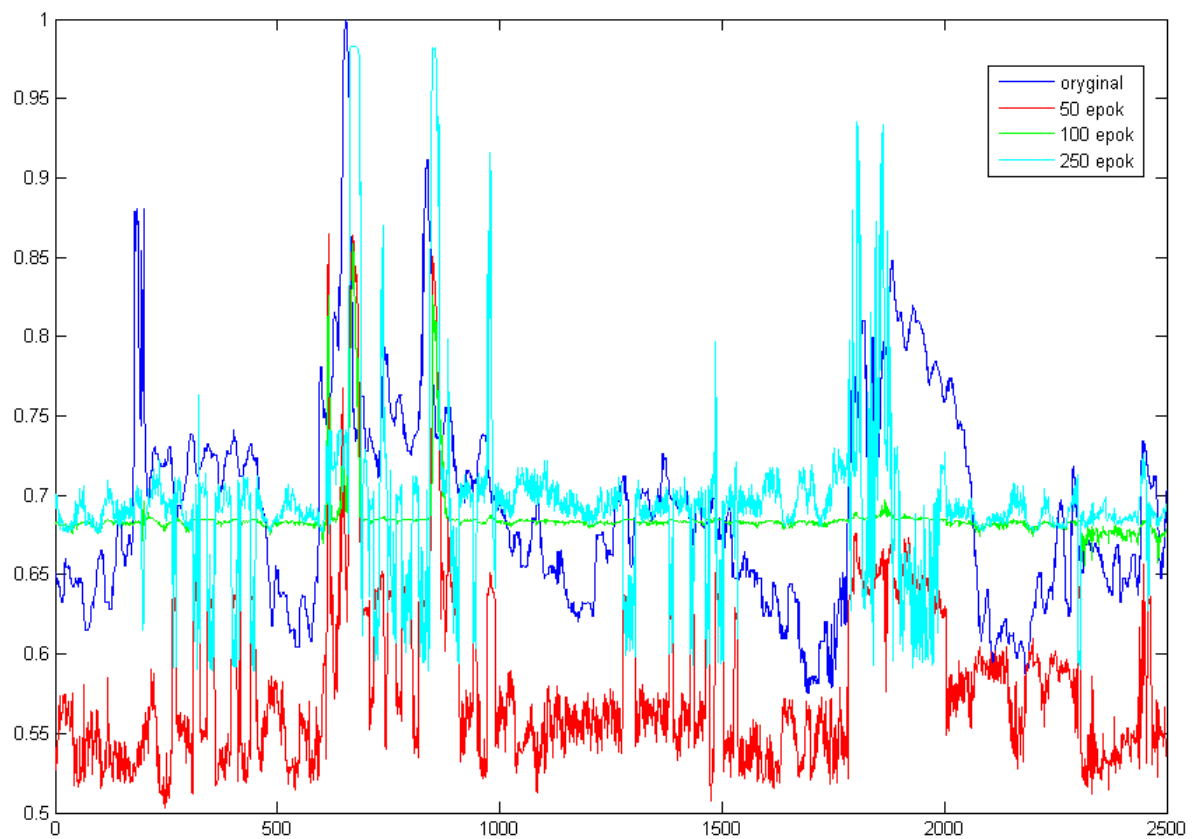
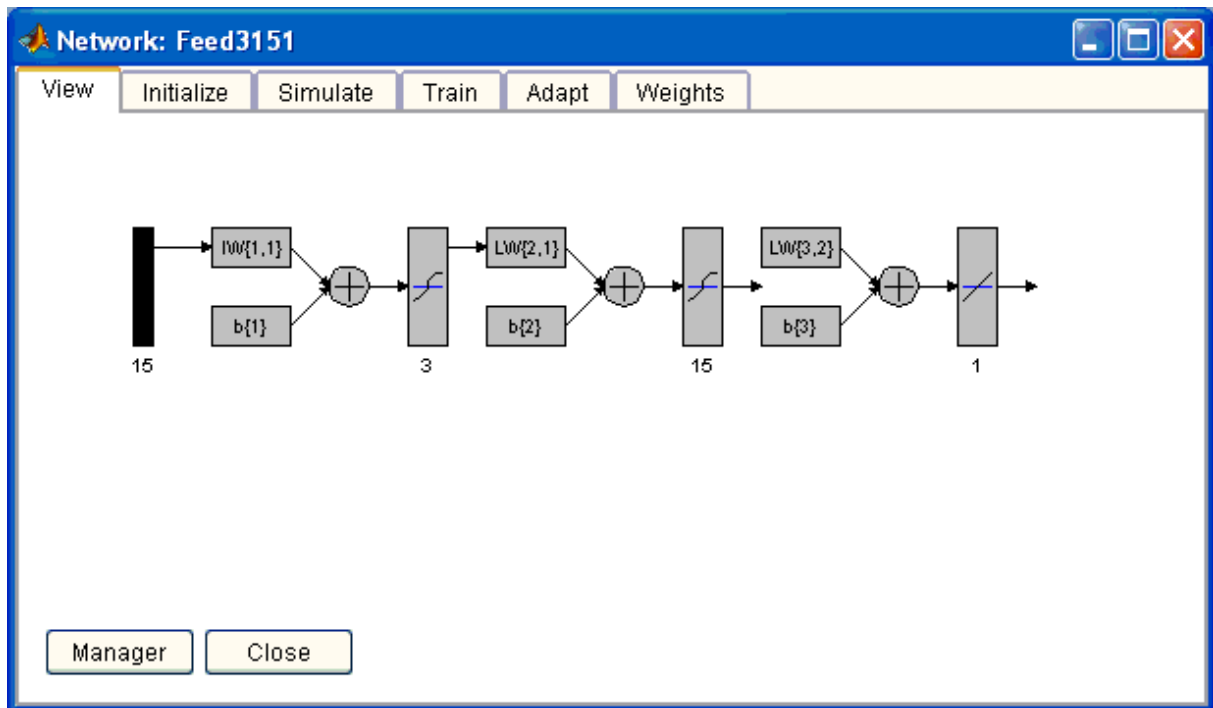
2.1.7 Trzy warstwy ukryte z liczbą neuronów 3-3-1



2.1.8 Trzy warstwy ukryte z liczbą neuronów 3-7-1

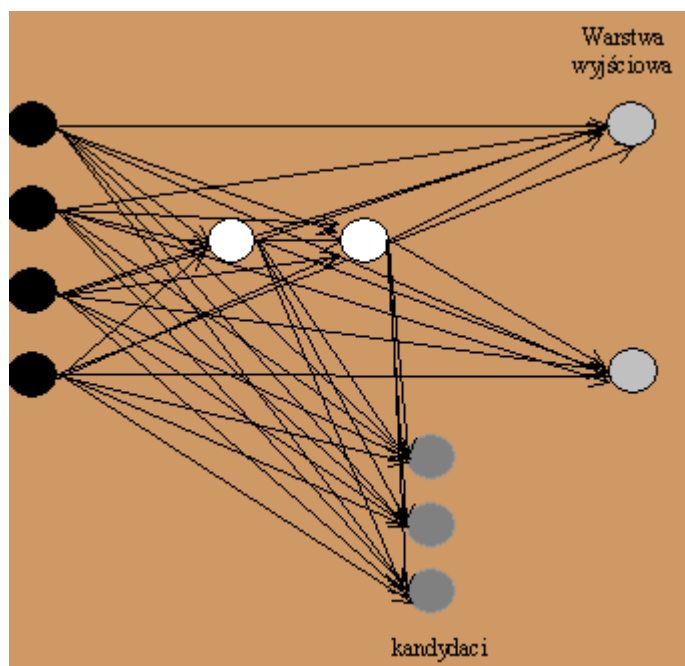


2.1.9 Trzy warstwy ukryte z liczbą neuronów 3-15-1



2.2 Sieci Cascade-forward backpropagation

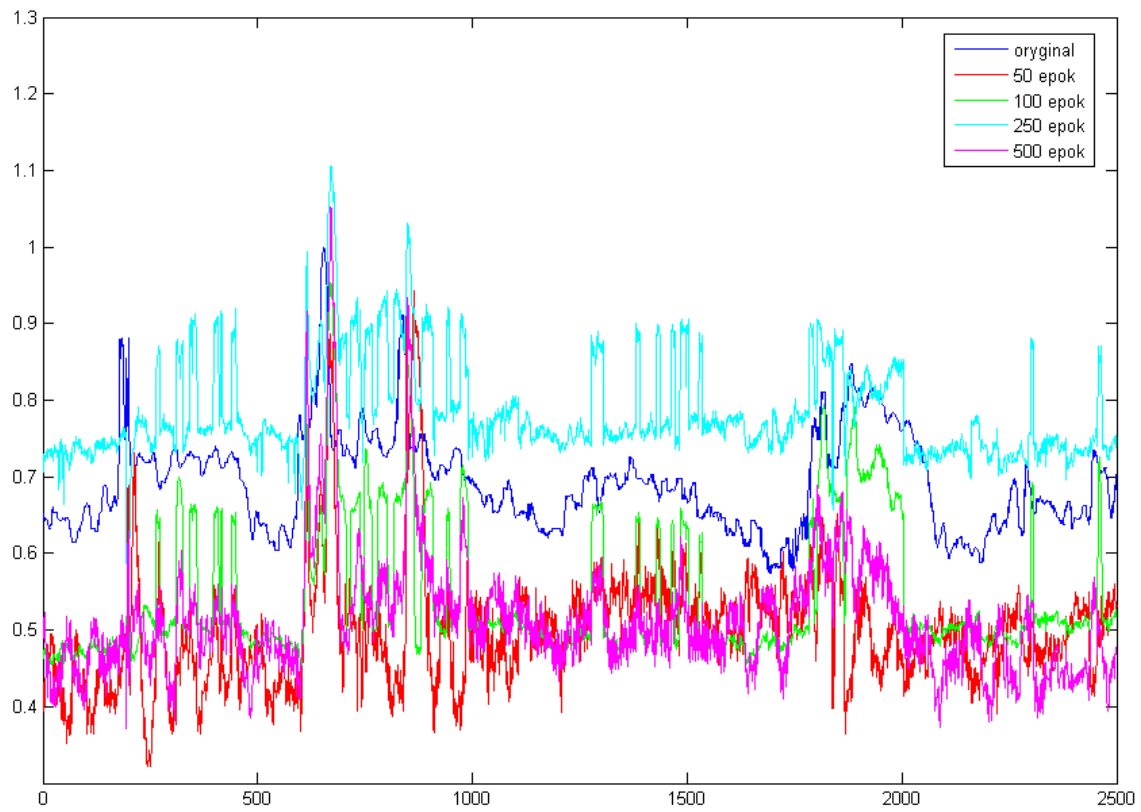
Krótki opis sieci:



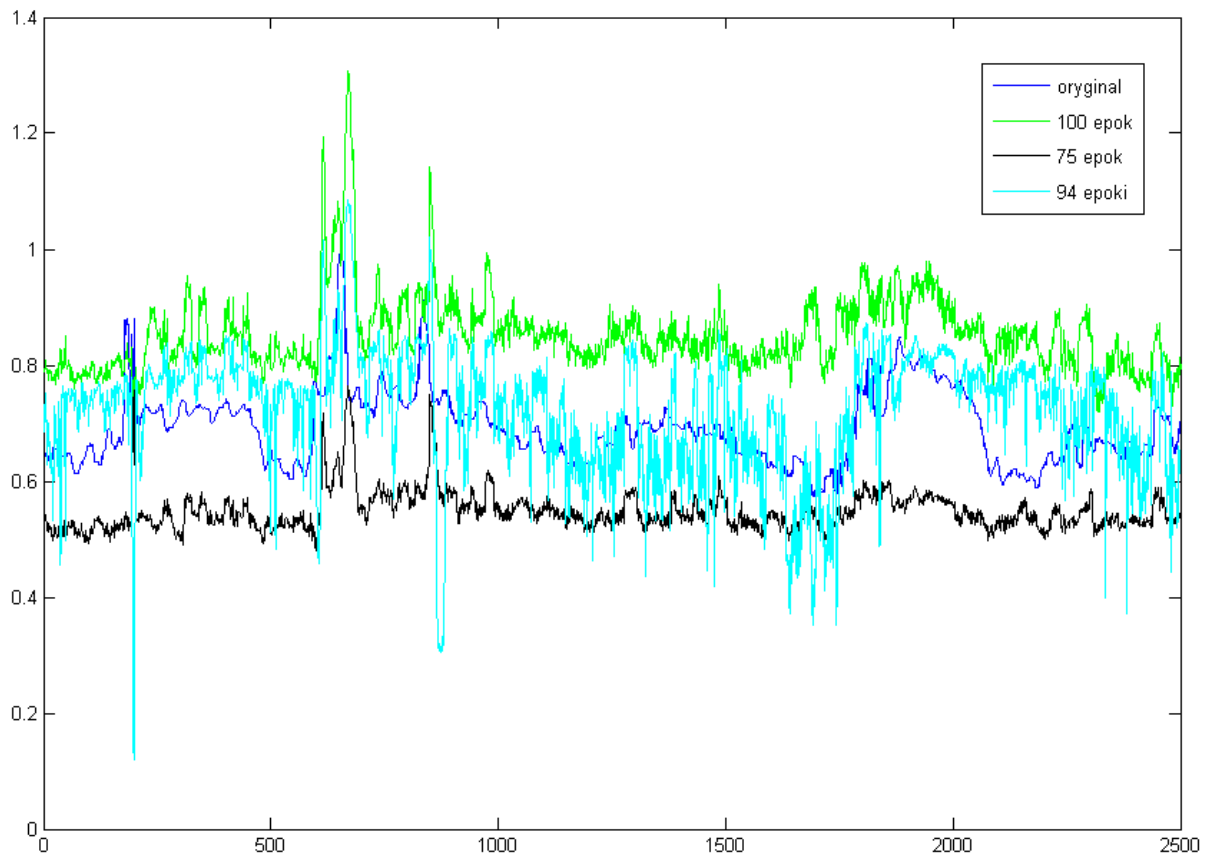
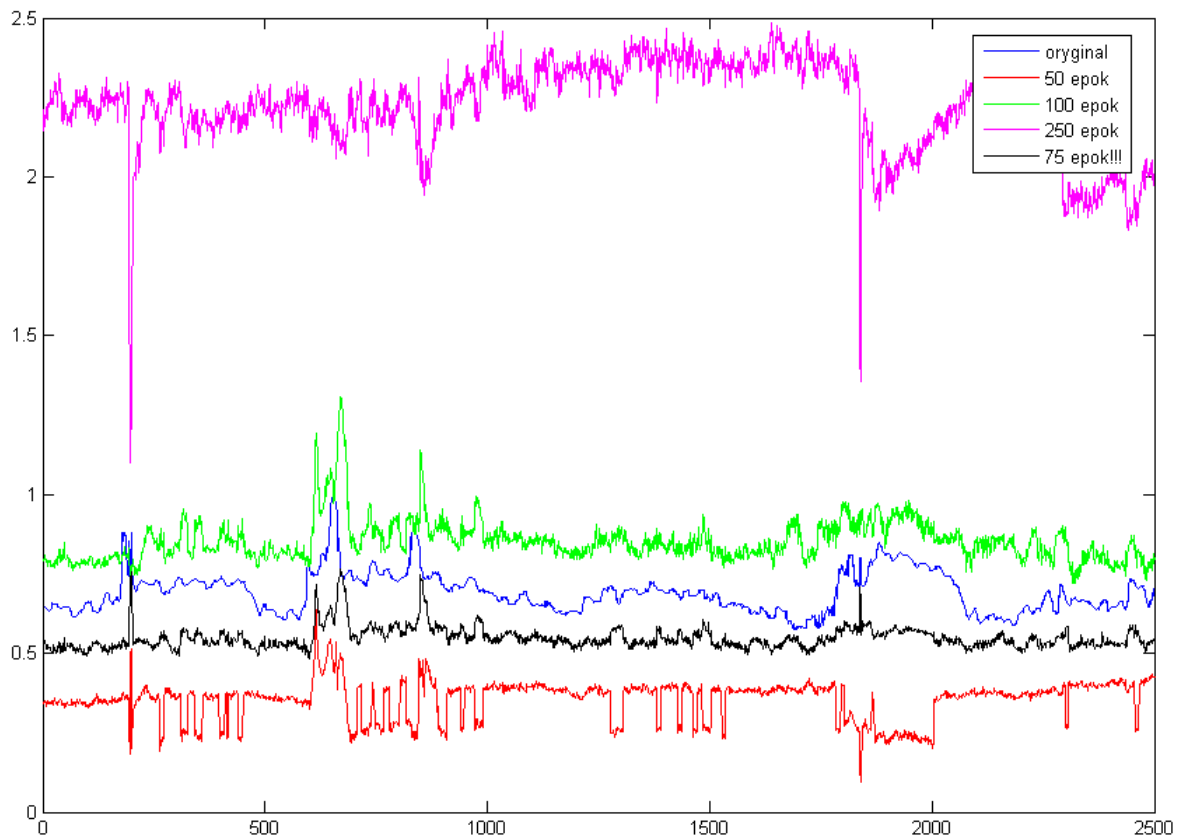
18. Rysunek obrazujący działanie sieci

- ✓ Sieć jednokierunkowa
- ✓ Metodą uczenia może być propagacja błędu wstecznego
- ✓ Po dodaniu kandydata, stare neurony ukryte mają ustalone wagi wejściowe
- ✓ Połączenia neuronów są w postaci rozwijającej się kaskady połączeń wagowych. Kolejno dokładany neuron ma połączenia z węzłami wejściowymi i wszystkimi już istniejącymi neuronami ukrytymi
- ✓ Następuje minimalizacja różnicy: $|E(\text{sieć}) - E(\text{kandydat})|$

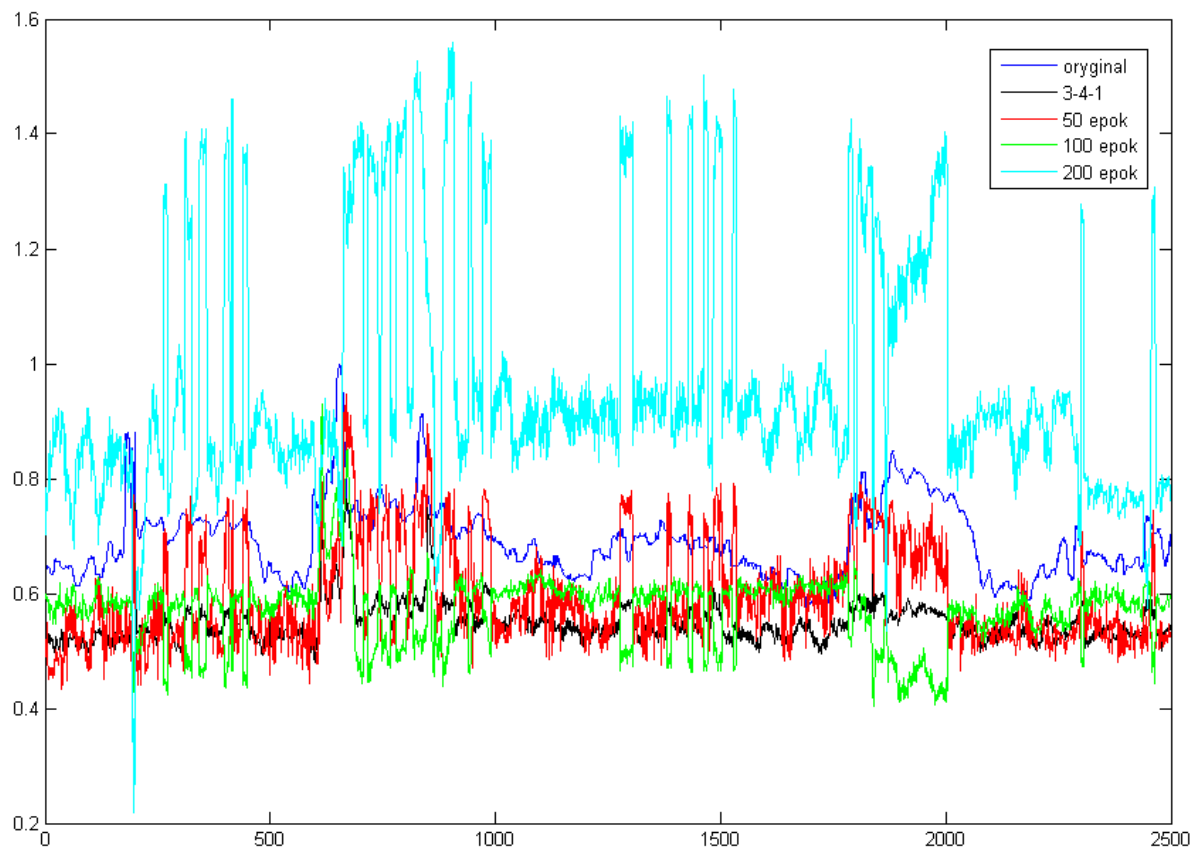
2.2.1 Trzy warstwy ukryte z ilością neuronów 3-2-1



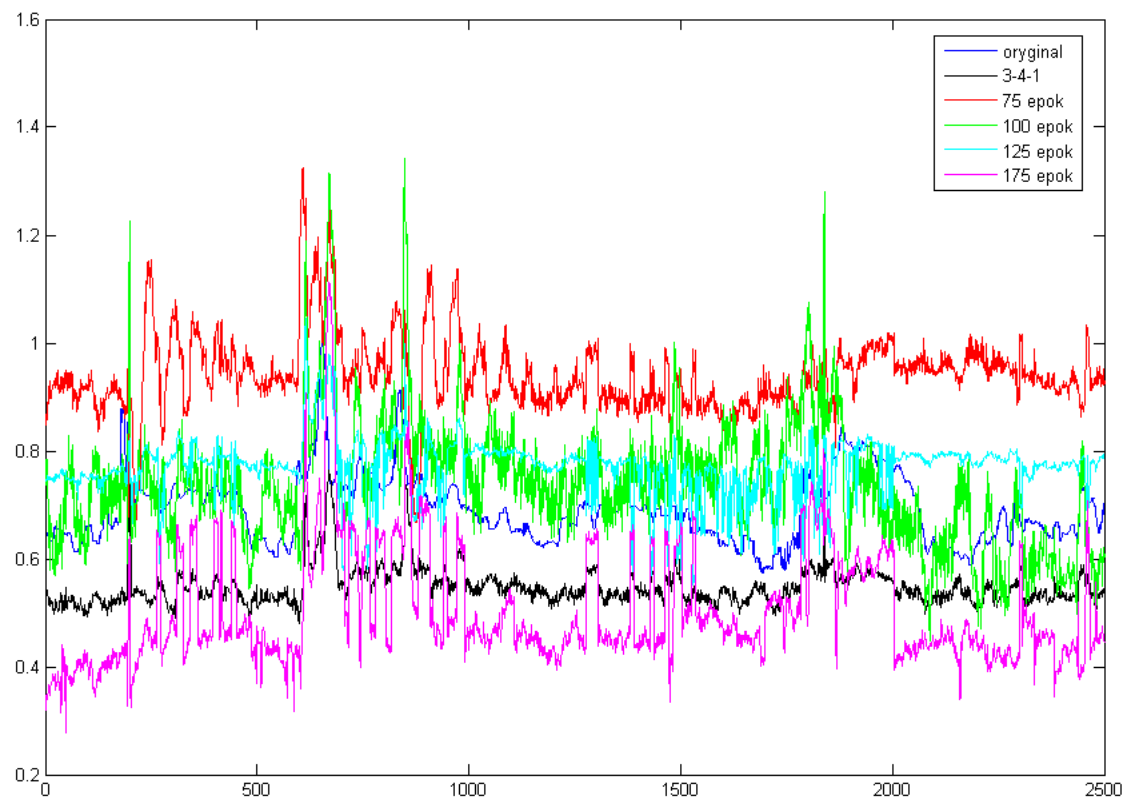
2.2.2 Trzy warstwy ukryte z ilością neuronów 3-4-1



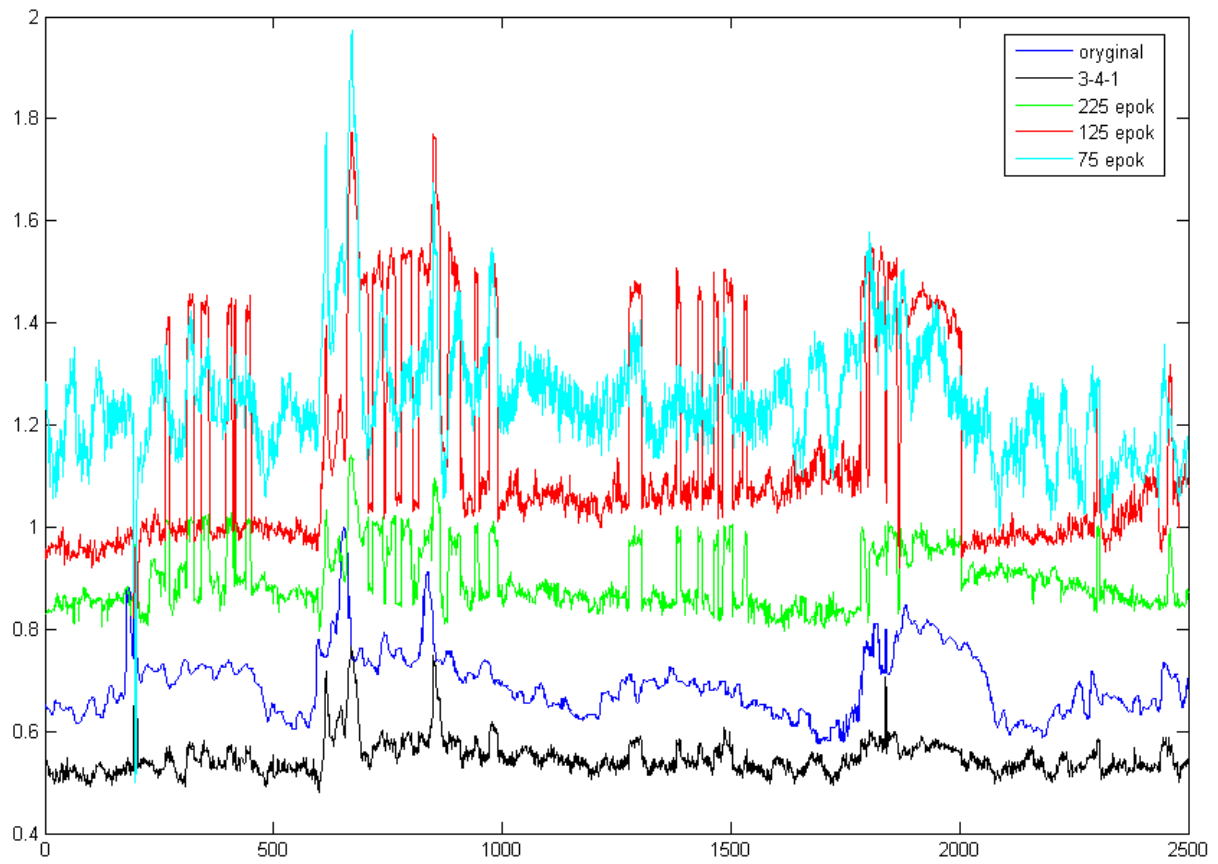
2.2.3 Trzy warstwy ukryte z ilością neuronów 5-3-1



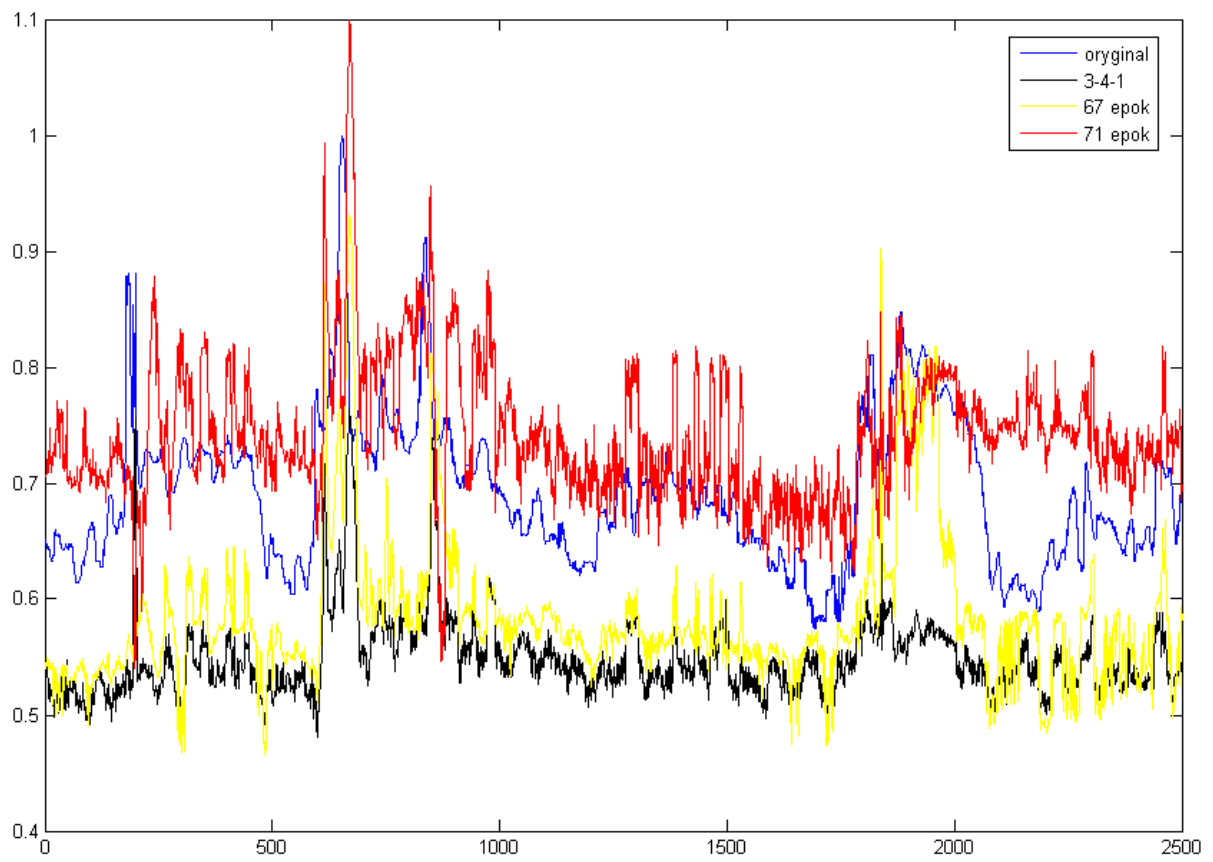
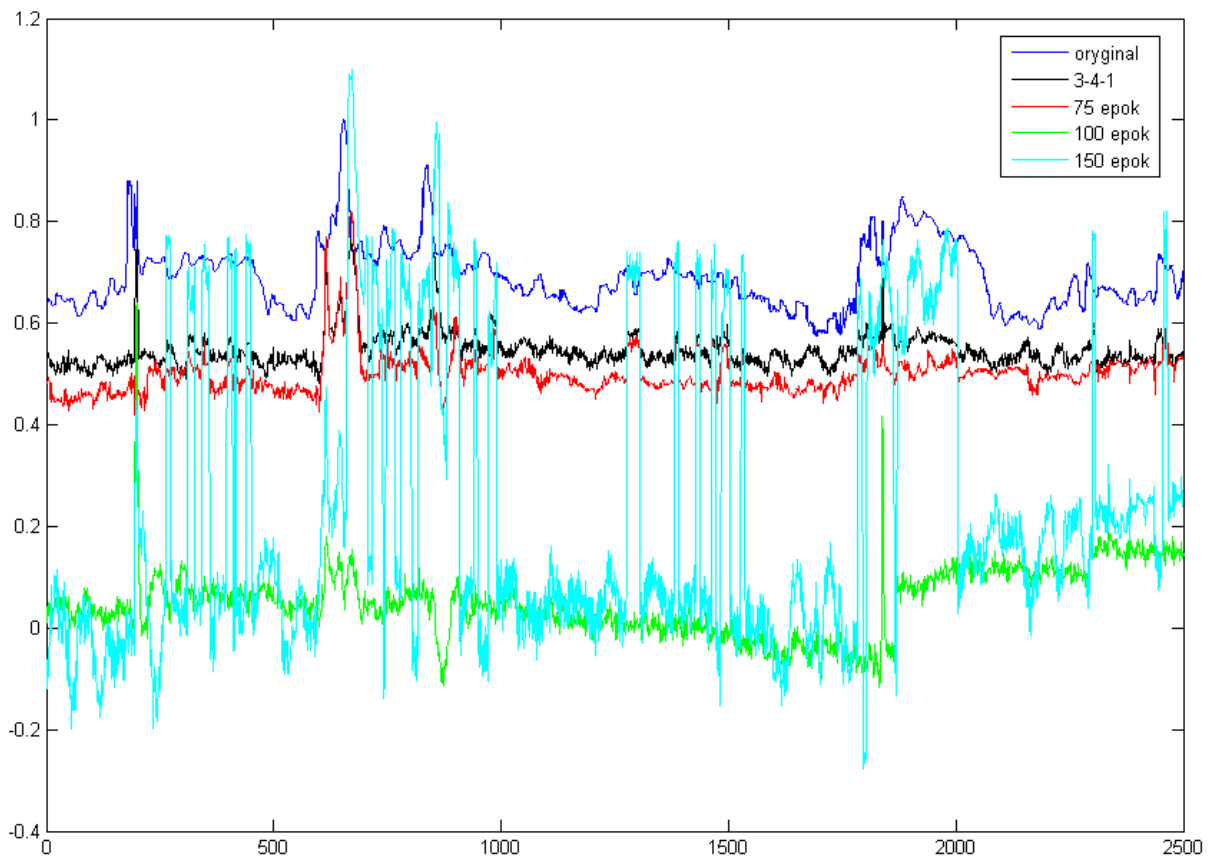
2.2.4 Trzy warstwy ukryte z ilością neuronów 4-4-1



2.2.5 Trzy warstwy ukryte z ilością neuronów 4-7-1

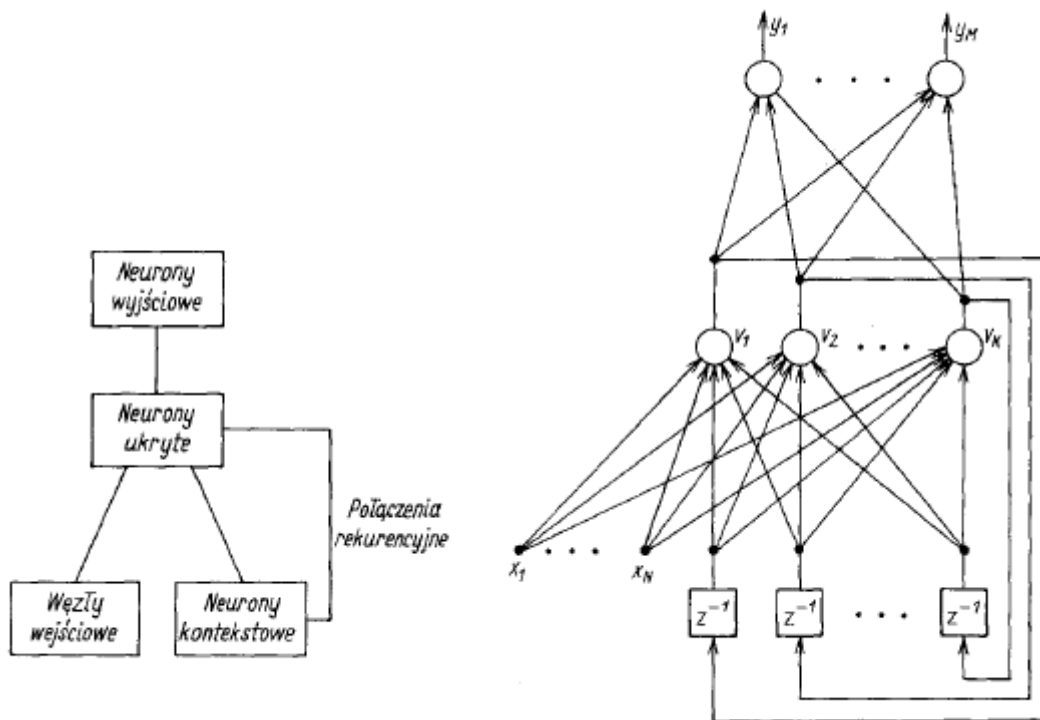


2.2.6 Dwie warstwy ukryte z ilością neuronów 4-1



2.3 Sieci Elman backpropagation

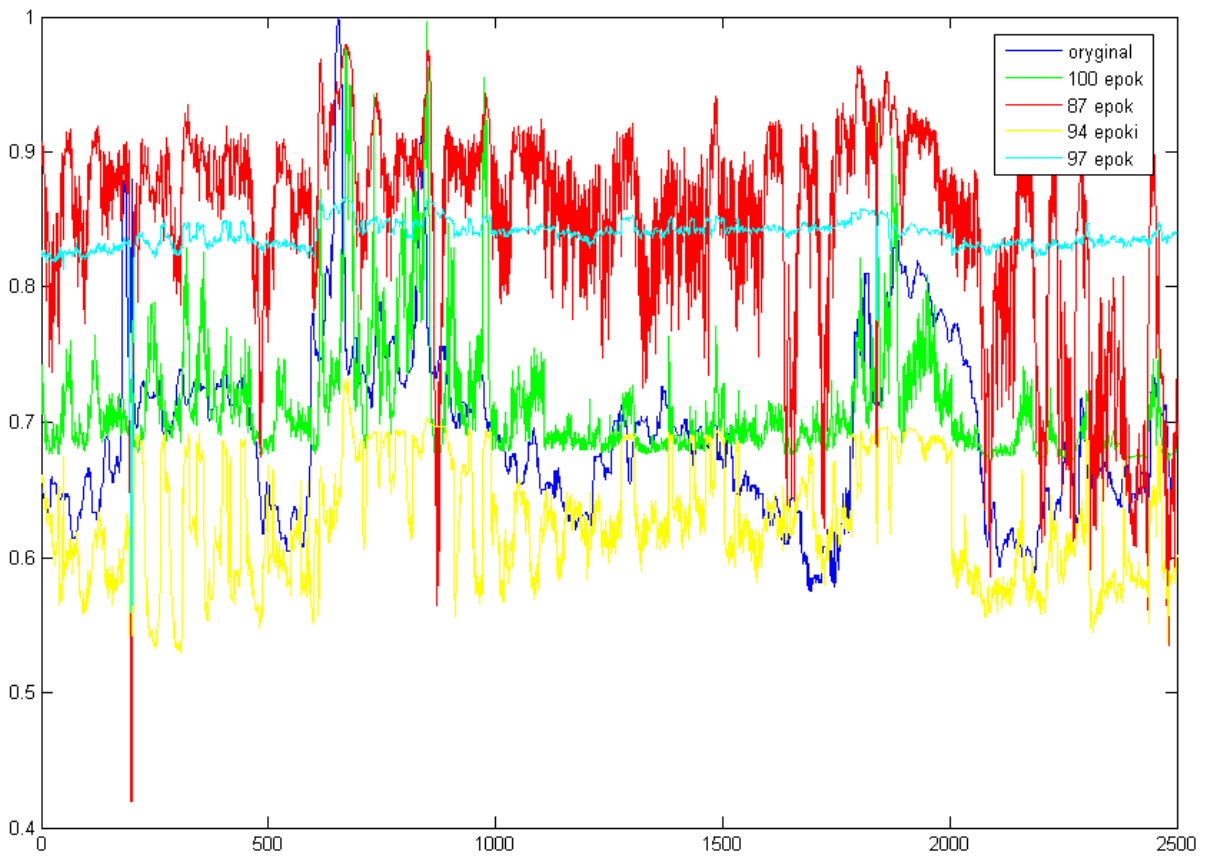
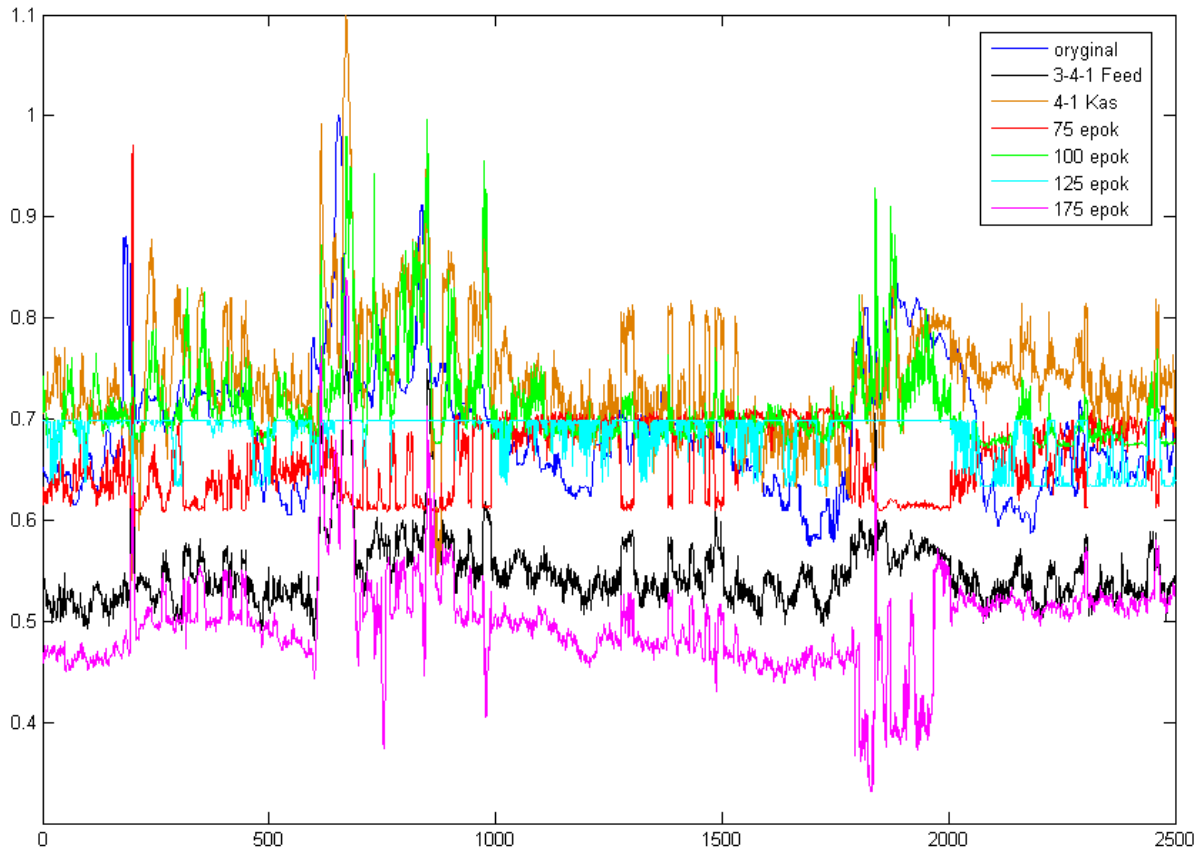
Krótki opis:



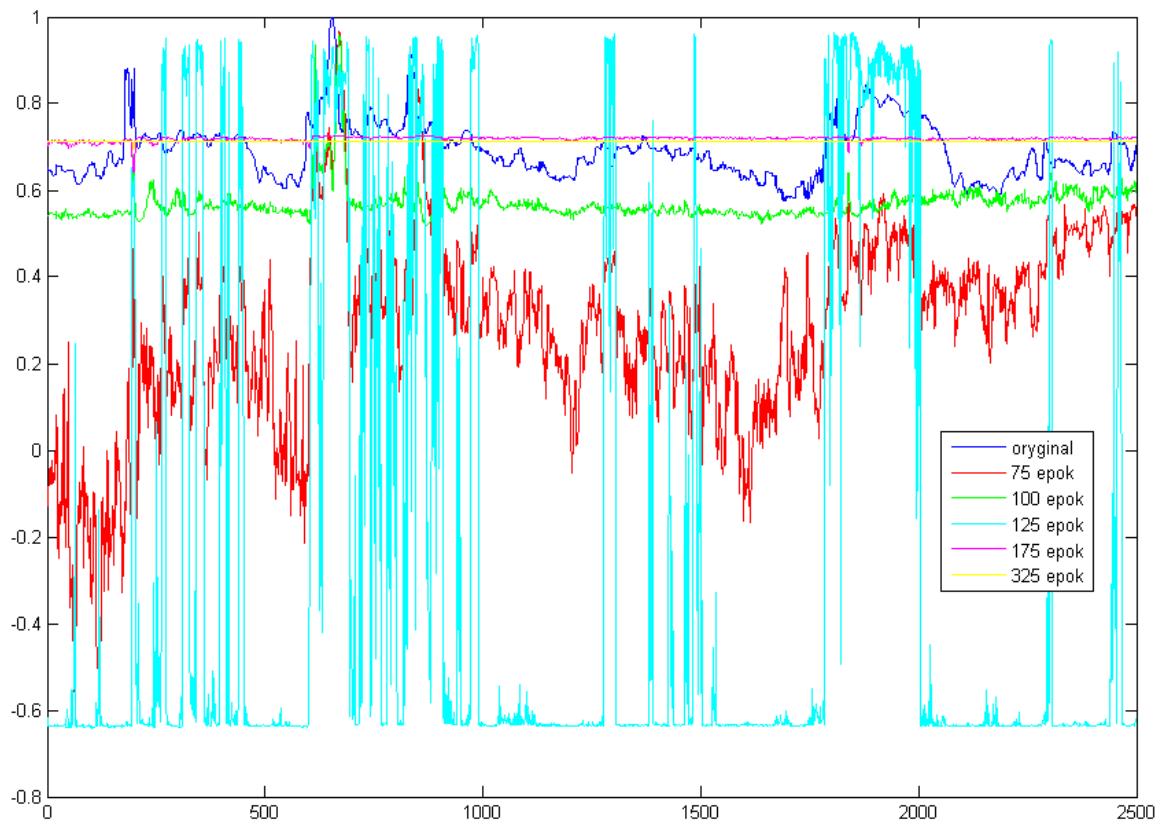
19. Postać ogólna wraz z układem połączeń

- ✓ Sieć ta jest częściowo rekurencyjna o strukturze dwuwarstwowej, a sprzężenie dotyczy tylko tej warstwy ukrytej
- ✓ Neurony warstwy wyjściowej są połączone tylko z neuronami warstwy ukrytej, czyli jest to sieć jednokierunkowa
- ✓ Każdy neuron ukryty ma swego odpowiednika w tzw. warstwie kontekstowej

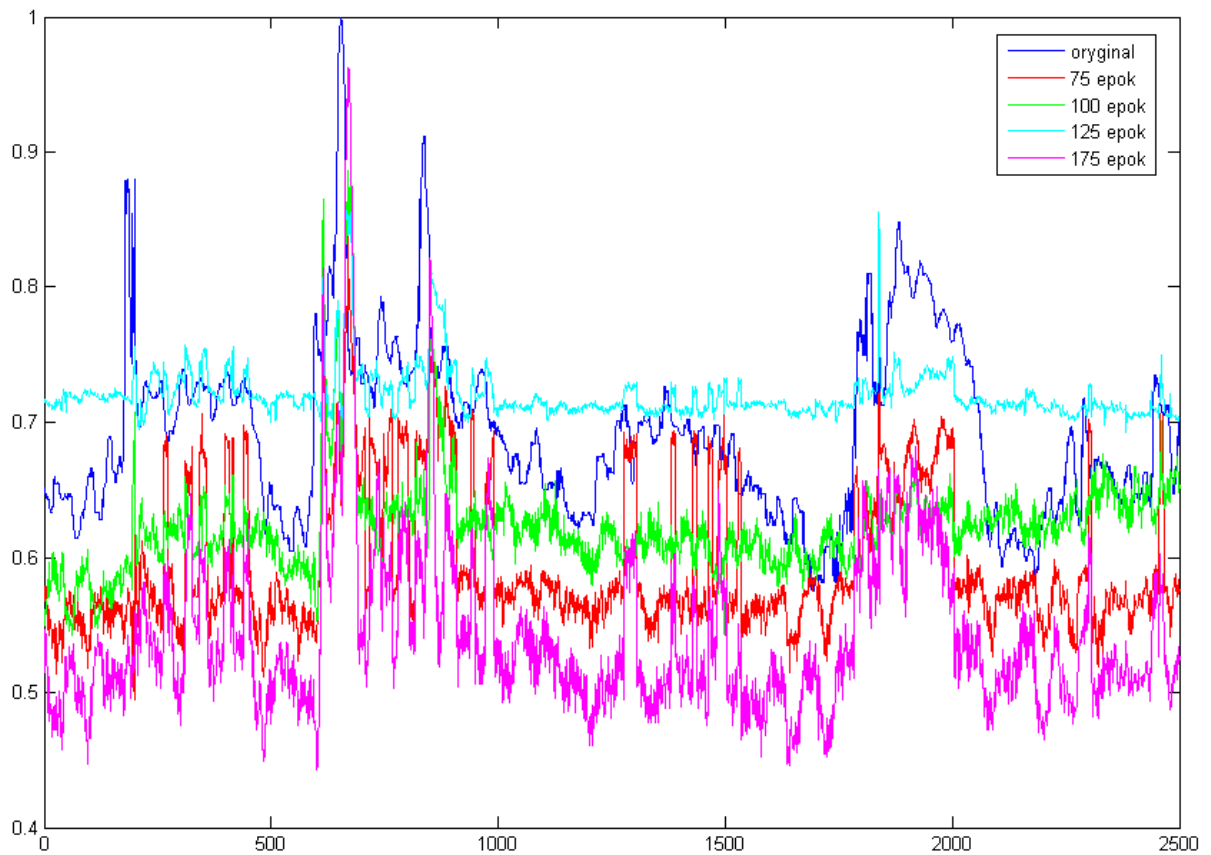
2.3.1 Dwie warstwy ukryte z ilością neuronów 4-1



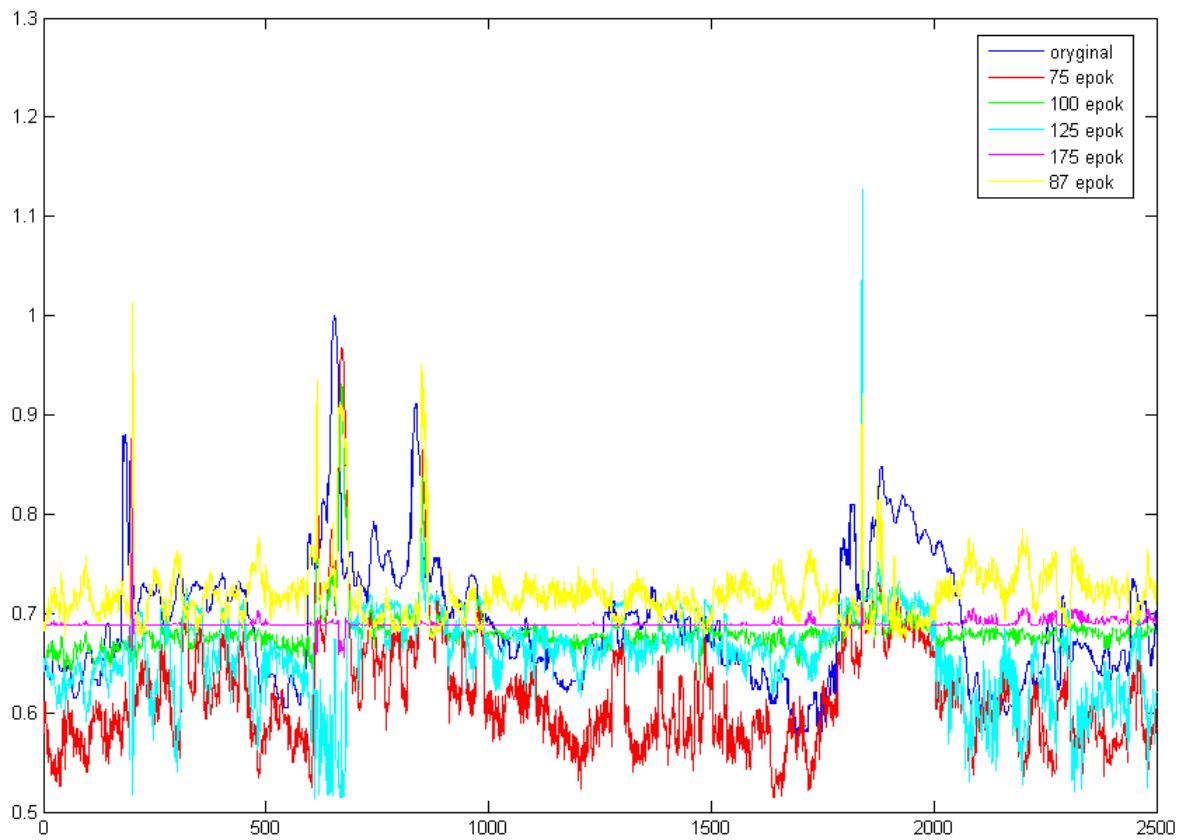
2.3.2 Trzy warstwy ukryte z liczbą neuronów 3-4-1



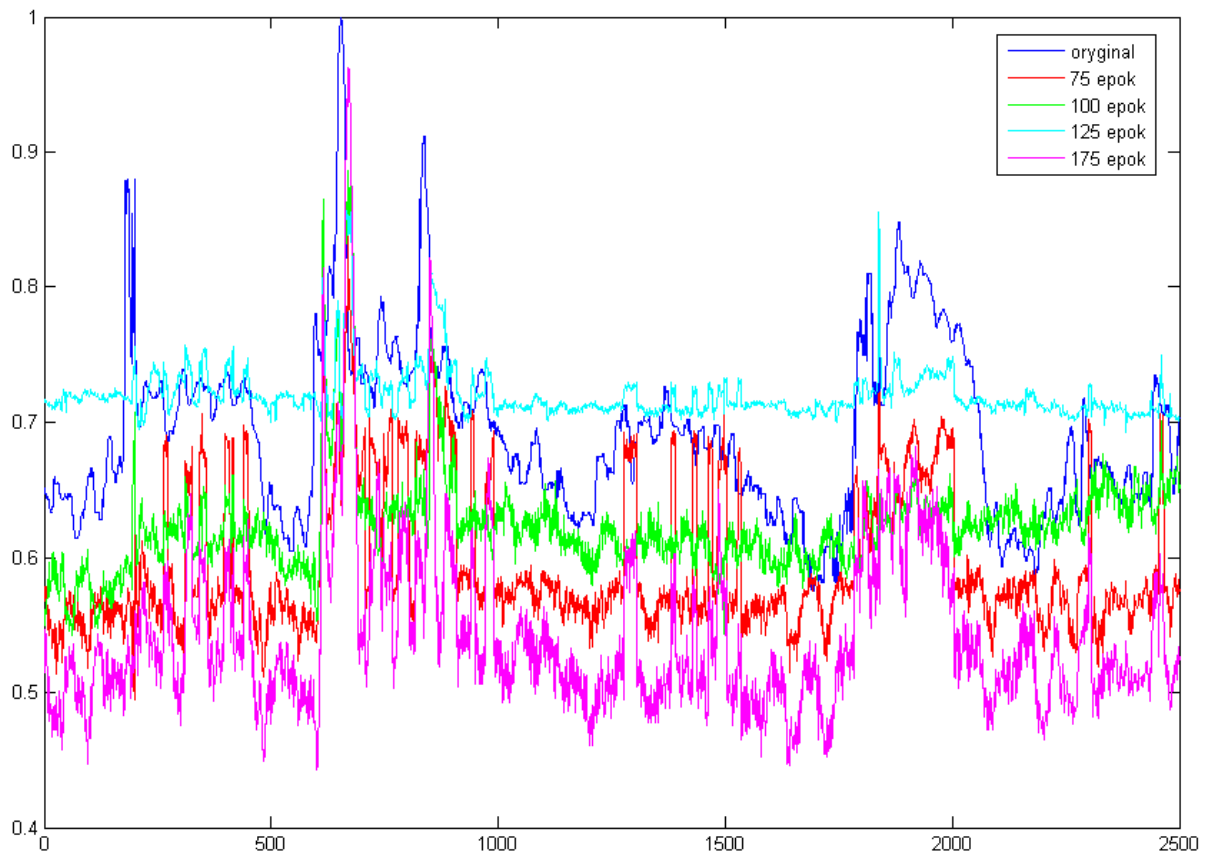
2.3.3 Trzy warstwy ukryte z liczbą neuronów 5-3-1



2.3.4 Trzy warstwy ukryte o liczbie neuronów 4-5-1

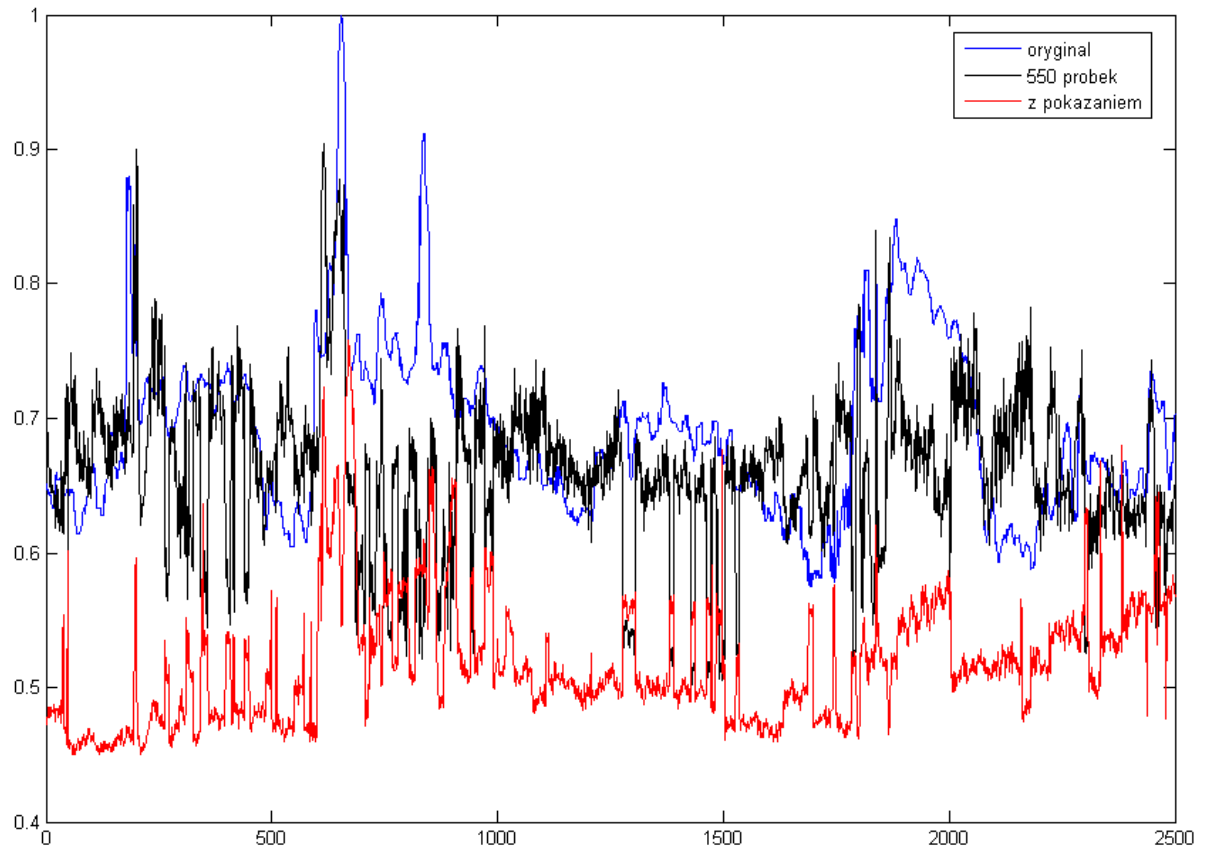


2.3.5 Trzy warstwy ukryte z liczbą neuronów 5-3-1

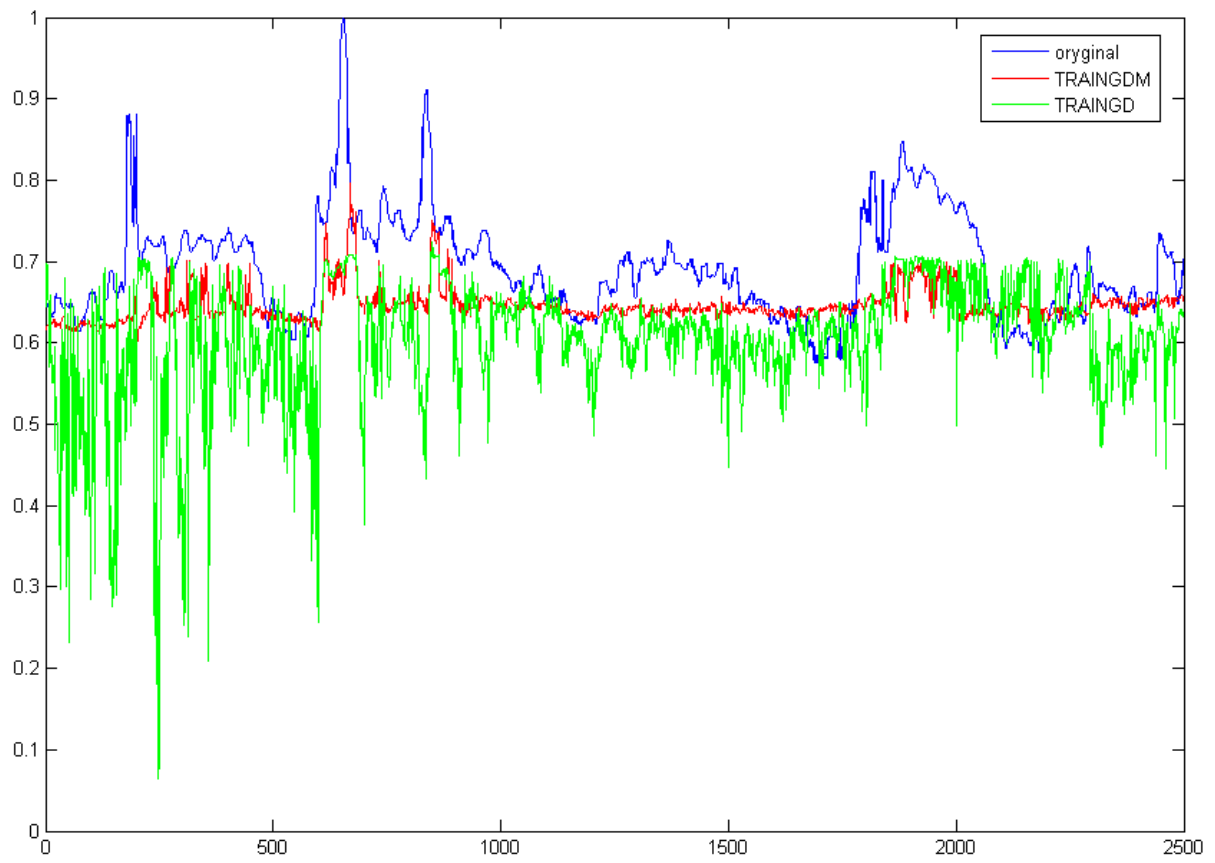


3 Eksperymenty z sieciami

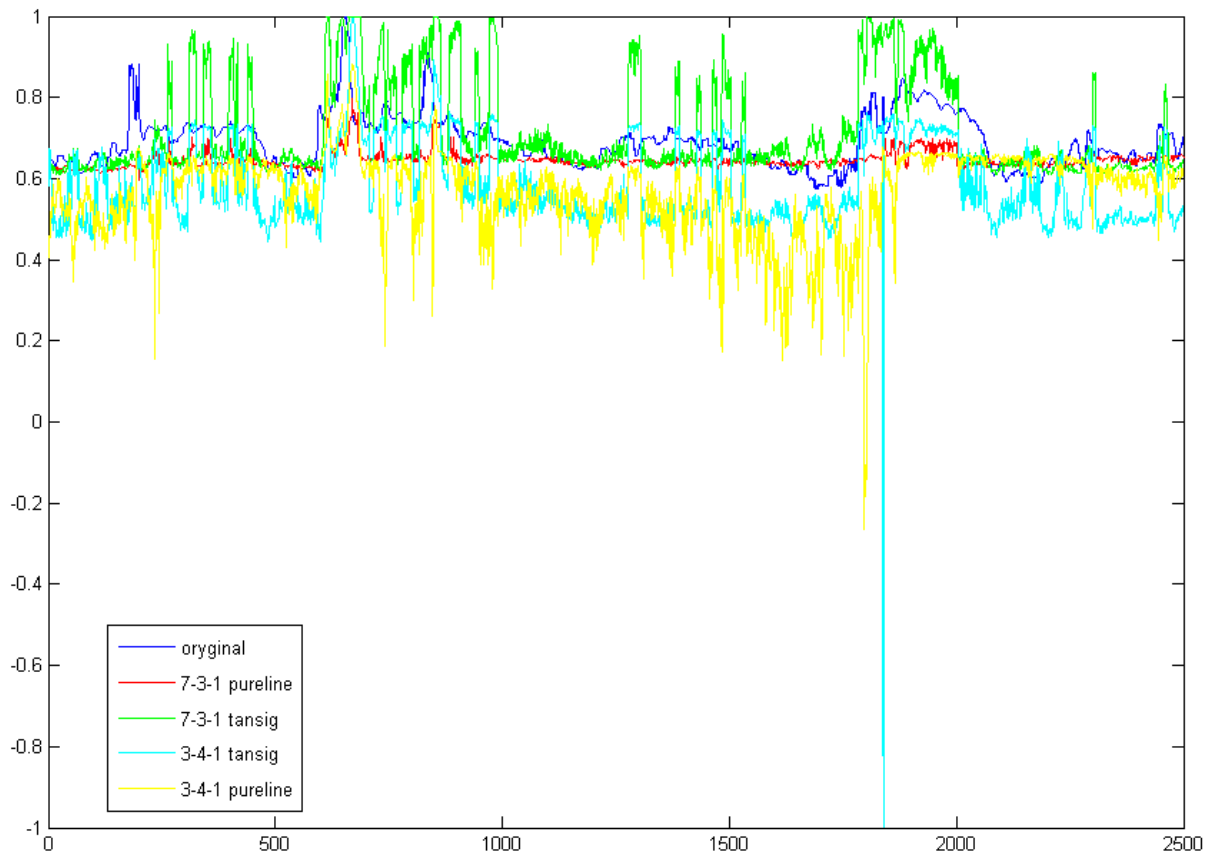
3.1 Pokazanie sieci podczas symulacji próbek, które są oczekiwane na wyjściu



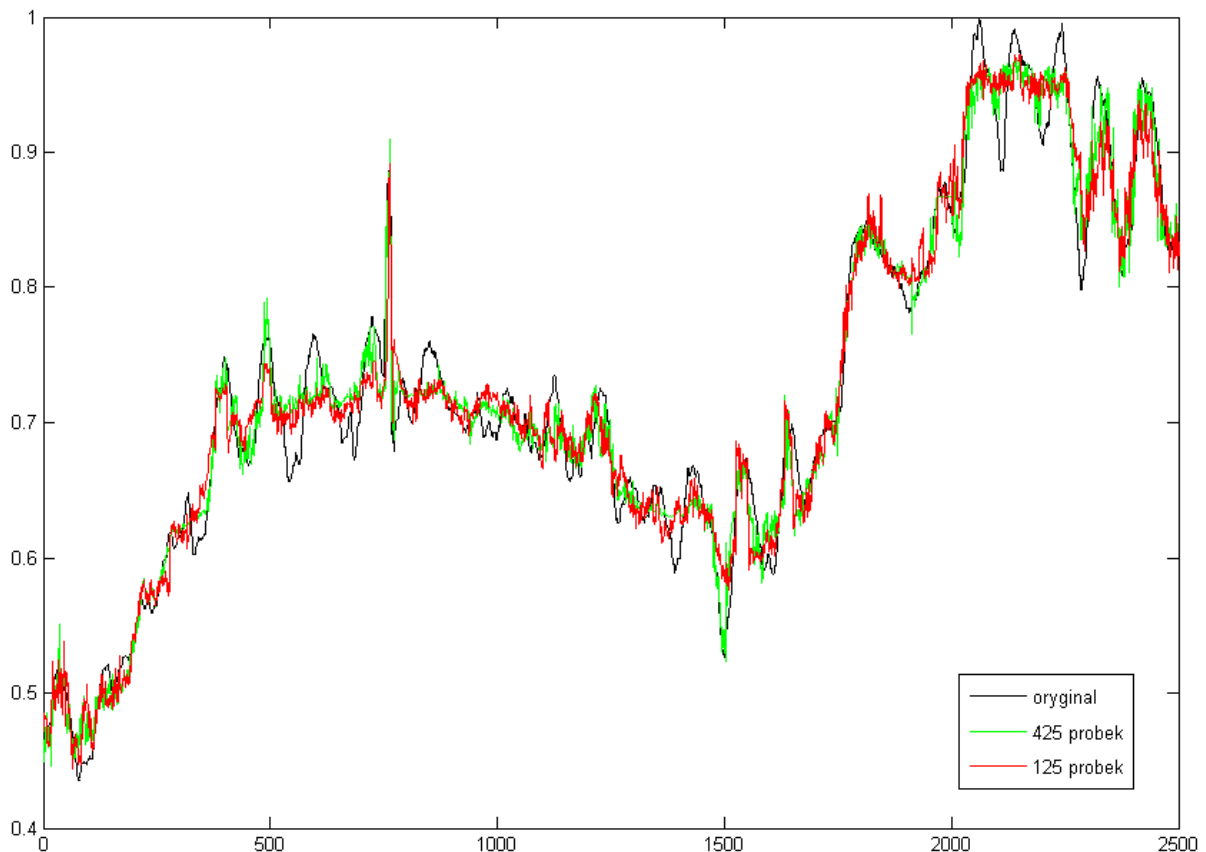
3.2 Zmiana funkcji adaptacji sieci



3.3 Zmiana funkcji przejścia w ostatniej warstwie



3.4 Ten sam zbiór danych dla uczenia i testowania



4 Analiza i wnioski z otrzymanych wyników badań

Zbadaliśmy wiele konfiguracji sieci neuronowych. Zwykle były to zmieniające się liczby neuronów w poszczególnych warstwach ukrytych oraz liczba epok przedstawiania sieci wyników z obiektu rzeczywistego, by sieć mogła zoptymalizować wagi, aby jak najlepiej poradzić sobie z postawionym problemem. Zmienialiśmy również konfigurację sieci, nie modyfikując przy tym funkcji przejścia. Dla wszystkich przebadanych sieci ustalaliśmy funkcję przejścia w ostatniej warstwie na linię prostą. Badania wykonywaliśmy zgodnie z procedurą opisaną w punkcie pierwszym naszej pracy, czyli obrabialiśmy wstępnie dane w programie Microsoft Excel, a następnie przy użyciu programu Matlab 7.0 uczyliliśmy sieć, a następnie symulowaliśmy jej działanie za pomocą próbek sprawdzających. Zauważyliśmy występowanie niekiedy anomalii, gdzie w procesie uczenia przy

pokazaniu sieci próbek symulacyjnych, ta zwracała w pewnym wąskim zakresie wynik, którego błąd rzędu kilkuset procent. Pomocna w naszej pracy okazała się gałąź nauki zwana metrologią, z której zaczerpnęliśmy wzory na odchylenie standardowe typu A, którego wyniki zobrazowane będą dla najlepiej i najgorzej nauczonej sieci.

Bardzo długo uczyła się sieć 15-1 dla 500 epok. Zwykle sieć ucząca się 50 epok jest zdecydowanie niedouczona (objawia się to niereagowaniem na sygnały na wejściu), natomiast sieci powyżej 200 epok wykazują tendencję do przeuczenia. Objawia się to wizualnie w postaci bardzo poszarpanego wykresu i dodawania pofałdowania pomiędzy próbkami, które były badane przez sieć. Ogólnie można zauważyć, iż sieci Feed-forward lepiej się spisują, gdy liczba neuronów w poszczególnych warstwach nie zmienia się w sposób drastyczny. Nie ma też sensu ustalanie zbyt wielkiej liczby warstw ukrytych, ani zbyt dużej liczby neuronów w poszczególnych warstwach, gdyż wyniki wcale nie są lepsze, a niekiedy gorsze niż te, jakich byśmy oczekiwali. Ponadto czas uczenia się sieci zwiększa się i trzeba długo czekać na wyniki symulacji nawet, gdy sprzęt, który jest wykorzystywany do badań, nie odbiega od obecnych standardów mocy obliczeniowej. W pracy przyjęliśmy również, że zgodnie z zaleceniami Prowadzącego projekt, nie przekraczamy liczby neuronów powyżej liczby 15, co jest również ilością wejść do naszej sieci neuronowej.

Zadowolające efekty wydaje się mieć sieć kaskadowa o ilości neuronów 3-4-1. Widać na wykresie, iż przebieg jest dość dobrze odwzorowany. Zwiększając wzmocnienie na wyjściu układu, można uzyskać przebieg zbliżony do oczekiwanego przez nas.

Wszystkie sieci mają problem z ustaleniem poprawnej decyzji w jednym fragmencie przedstawionego procesu. Wskazuje to, iż w próbkach uczących, ten zakres zmienności funkcji nie występuje lub jest go na tyle mało, że sieci nie potrafią się poprawnie reagować w przyszłości, mając do czynienia z podobnym problemem.

Próbowaliśmy pokazać sieciom podczas symulacji również wyniki, jakie powinna uzyskać (targets). Okazało się to ślepym zaułkiem. Wyniki wcale nie były lepsze, a w większości przypadków badanych okazało się, iż otrzymane dane odbiegają w bardziej znaczący sposób od wyników uzyskiwanych dla sieci bez pokazywania wyjścia.

Próba zmiany funkcji adaptacji sieci z domyślnej tzn. LEARNGDM na LEARNGD, także nie daje pozytywnych rezultatów. Jak pokazane dla jednego z przebiegów, wyniki dla tej drugiej osiągają wartości o dużym zakresie zmian w krótkim okresie czasu. Innymi słowy, sieć zmienia gwałtownie wartości na swoim wyjściu nie kontynuując trendu, jaki zdaje się wyznaczać funkcja rzeczywista.

Wnioskiem, który zdaje się być najbardziej istotny po przeprowadzonych badaniach jest to, iż najlepiej uczą się sieci o średniej ilości neuronów w poszczególnych warstwach, tzn. kształtujących się na poziomie od trzech do pięciu neuronów. Potwierdza się to dla różnych typów sieci neuronowych przez nas zbadanych. Kolejnym wnioskiem jest to, że sieci o dwóch warstwach ukrytych dość dobrze radzą sobie z problemem, który przed siecią postawiliśmy. Również tutaj, polecamy średnią ilość neuronów. Sprawdziliśmy dla 15 neuronów i wyniki były niezadowolające, co zresztą napisaliśmy kilka akapitów wcześniej. Ponadto zmiana funkcji przejścia w ostatniej warstwie

ukrytej skutkowała kompletnie błędnym uczeniem sieci, więc jako funkcję przejścia dla warstwy ostatniej polecamy $f(x)=x$.

Doskonale odwzorowane są przebiegi, gdy podczas symulacji zamiast podawać próbki testujące, podajemy po raz kolejny próbki uczące. Wykres jest wówczas dość dobrze odwzorowany, a po obliczeniu błędu, wynik był równy 1,136%. Dla innych konfiguracji sieci błędu nie liczyliśmy, gdyż jest on tak duży, że nie ma sensu go liczyć i porównywać dokładnie wyników. Większość wyjdzie zbliżonych, a zadowalające efekty porównania sieci są już, gdy błąd zobrazowany jest graficznie w postaci wykresów dla każdej z konfiguracji.

Wnioskiem ostatecznym, jaki nasuwa się po kilkudziesięciu godzinach pracy nad znalezieniem idealnej sieci jest następujący: Należy pogodzić się z dużym błędem w estymacji sygnału przez sieć. Wybór sieci powinien paść na taką, która ma średnią liczbę neuronów i oferuje kompromis pomiędzy szybkością liczenia, a błędami. Widać było, iż te zmieniały się zaledwie w granicach kilku procent dla naszych „średniaków”.