ZASTOSOWANIE SZTUCZNYCH SIECI NEURONOWYCH W ANALIZIE SYGNAŁÓW ELEKTROKARDIOGRAFICZNYCH

USING ARTIFICIAL NEURAL NETWORKS FOR ANALYSIS OF ELECTROCARDIOGRAPHIC SIGNALS

Monika Litwińska*

Politechnika Gdańska, Wydział Mechaniczny, 80-233 Gdańsk, ul. Narutowicza 11/12

***** e-mail: monika.litwinska@wp.pl

STRESZCZENIE

Celem pracy było przebadanie możliwości zastosowania sztucznych sieci neuronowych do analizy i rozpoznawania sygnałów EKG. Artykuł zawiera przegląd zagadnień dotyczących EKG, pozyskiwania i interpretacji sygnałów oraz zastosowania sztucznych sieci neuronowych do diagnostyki. Znaczącym elementem pracy jest próba zaimplementowania w programie Matlab systemu rozróżniającego sygnały różnego typu.

Słowa kluczowe: EKG, sztuczna inteligencja, sztuczne sieci neuronowe, Matlab

ABSTRACT

The main goal of the work is to test the possibility of using artificial neural networks for analysis and recognition of ECG signals. The paper contains a review of issues related to ECG, acquisition and interpretation of signals, application of artificial neural networks in the diagnosis. A significant element is the attempt to implement the system for differentiating various types of signals using Matlab software.

Keywords: ECG, artificial intelligence, artificial neural networks, Matlab

1. Wstęp

Wiele zagadnień i problemów, które potrafi rozwiązać człowiek, jest trudnych do analizowania, gdyż nie udaje się znaleźć reguł, które można zapisać jako prosty algorytm. Takie zagadnienia jak rozpoznawanie twarzy, czy rozpoznawanie obrazów, kojarzenie faktów w oparciu o ich ogólne podobieństwo, są dla człowieka łatwe, natomiast trudne do zaprogramowania w komputerze. Za podejmowanie decyzji oraz rozumowanie u człowieka odpowiedzialny jest mózg. Składa się on z wielu różnych elementów, jedną z najistotniejszych jego części jest neuron. Mózg zawiera ich około 10^{11}, a 10^{15} połączeń pomiędzy neuronami tworzy sieć neuronową. Neurony są nieliniowymi jednostkami przetwarzania sygnałów i w efekcie, mózg podejmujący decyzje, działa jak bardzo skomplikowana i rozbudowana sieć odpowiednio połączonych neuronów. Fakt ten dał impuls do rozwoju sztucznych sieci neuronowych (SSN), gdzie algorytmy przetwarzania informacji są budowane w oparciu o sieci sztucznych neuronów, których właściwości odpowiadają neuronom

biologicznym [1].

Do typowych zagadnień wykonywanych przez człowieka w diagnostyce medycznej, a jak dotychczas nie dającym się w całkowicie zautomatyzować, należy rozpoznawanie sygnałów elektrokardiograficznych (EKG) w celu diagnostyki pracy serca. Dlatego też jednym z bardziej rozwijanych kierunków badawczych jest wykorzystanie sztucznych sieci neuronowych do analizy i rozpoznawania sygnałów EKG. Celem pracy było przebadanie możliwości zastosowania sztucznych sieci neuronowych do analizy i rozpoznawania sygnałów EKG. Znaczącym elementem była próba zaimplementowania w oprogramowaniu Matlab prostego systemu rozróżniającego prawidłowe i nieprawidłowe sygnały EKG przy pomocy sieci neuronowej.

2. Sztuczna inteligencja

Nie ma jednej właściwej definicji inteligencji, dlatego trudno zdefiniować inteligencję sztuczną. Pojęcie sztucznej inteligencji zrodziło się w 1956 roku, kiedy to John McCarthy zorganizował seminarium wakacyjne, w trakcie którego uczestnicy zastanawiali się, jakie aspekty działalności ludzkiej mogą zostać zastąpione przez komputer. Wówczas po raz pierwszy użyto terminu „sztuczna inteligencja" [2, 3].

Sztuczna inteligencja AI (ang. *Artificial Intelligence*) to dziedzina, która może zmienić całkowicie nasz świat. Są dwa odmienne spojrzenia na sztuczną inteligencję. Pierwsze z nich dotyczy tzw. słabej sztucznej inteligencji. Według tej definicji komputer pozwala formułować i sprawdzać hipotezy związane z mózgiem. Hipotezy te są jak najbardziej słuszne. Spotkać się można również z twierdzeniem, które mówi, że odpowiednio zaprogramowany komputer jest w istotny sposób równoważny mózgowi. W pracy [4] sieć neuronowa to urządzenie techniczne lub algorytm, którego działanie wzorowane jest w pewnym stopniu na działaniu sieci komórek biologicznych. Drugim rodzajem sztucznej inteligencji jest tak zwana silna sztuczna inteligencja. Postuluje ona możliwość zbudowania systemu zdolnego komunikować się z ludźmi bezpośrednio w języku naturalnym, rozumieć niuanse, przyjmować polecenia oraz planować ich realizację [1, 4]

Krótko mówiąc, sztuczna inteligencja to wspólną gałąź dwóch dyscyplin naukowych: informatyki oraz robotyki. Ogólnie jest też przedmiotem zainteresowania kognitywistyki – dziedziny nauki zajmującej się analizą działania zmysłów, mózgu i umysłu, w szczególności ich modelowaniem.

3. Sztuczne sieci neuronowe

Sieci neuronowe powstały w wyniku badań w dziedzinie sztucznej inteligencji. Główny wpływ na ich powstanie miały prace dotyczące struktur w mózgu. Prace te miały na celu naśladować cechy charakteryzujące biologiczne systemy nerwowe, czyli odporność na uszkodzenia oraz zdolność do uczenia się [3, 4].

Do tej pory nie wiadomo, w jaki sposób ludzki mózg uczy się i zapamiętuje informacje, ale naśladujące go sztuczne sieci neuronowe, które wzorowane są na prawdziwych komórkach nerwowych, potrafią zapamiętywać i uczyć się na podstawie wcześniej gromadzonych danych. Sztuczne sieci neuronowe mogą podejmować decyzje, jak również zastępować człowieka w skomplikowanych czynnościach wymagających nie tylko wiedzy, ale i intuicji [5].

Sztuczna sieć neuronowa to uproszczony, ale i bogaty model rzeczywistego biologicznego systemu nerwowego, inaczej mówiąc jest to uproszczony model ludzkiego mózgu. Składa się ona z dużej liczby elementów przetwarzających informacje – sztucznych neuronów. Taki sztuczny neuron ma zredukowane funkcje względem biologicznego odpowiednika, ograniczające się do wykonywania kilku najprostszych zadań [3, 5, 6, 7, 8].

Sztuczne sieci neuronowe są wygodnym jak i przydatnym narzędziem do realizowania wielu praktycznych zadań. Znalazły one zastosowanie w takich dziedzinach jak: finanse, medycyna, inżynieria, geologia czy fizyka. W rzeczywistości sieci znalazły zastosowanie wszędzie, gdzie ważne jest przetwarzanie, analiza danych, predykcja, klasyfikacja czy sterowanie. Sztuczne sieci neuronowe w rzeczywistości to techniki modelowania, które wykazują zdolność do odwzorowania złożonych funkcji. Ich najważniejszą zaletą jest kontrola nad problemem wielowymiarowości. Sieci neuronowe same konstruują potrzebne użytkownikowi modele, ponieważ automatycznie uczą się na podanych

przez niego przykładach. Krótko mówiąc, sztuczne sieci neuronowe odzwierciedlają działanie ludzkiego umysłu. Oparte są one na prostym modelu, przedstawiającym działanie biologicznego systemu nerwowego [5, 6].

3.1. Rodzaje sztucznych sieci neuronowych

Sposoby połączenia neuronów oraz ich współdziałanie przyczyniły się do powstania różnych rodzajów sieci. Na typ sieci ma wpływ kierunek przepływu sygnału w sieci. Każdy typ sieci charakteryzuje się własną metodą doboru wag, czyli sposobu uczenia. Istnieje bardzo wiele rodzajów sieci neuronowych, jednak najbardziej podstawowe, obrazujące budowę i sposób działania, to:

- sieci jednokierunkowe
  - jednowarstwowe
  - wielowarstwowe
- sieci rekurencyjne
- sieci komórkowe [9].

3.2. Proces uczenia

Sieci nie są tworzone w sposób typowy dla klasycznych programów komputerowych. Istnieją metody uczenia i samouczenia pozwalające uzyskać działanie nawet wówczas, kiedy twórca nie zna algorytmu, według którego można rozwiązać postawione zadanie [7, 10, 11].

Wyróżnia się dwa sposoby uczenia sieci: uczenie z nauczycielem – uczenie nadzorowane (ang. *supervised learning*) oraz uczenie bez nauczyciela – uczenie nienadzorowane (ang. *unsupervised learning*). Elementem różnicującym powyższe sposoby uczenia jest struktura zbioru uczącego. Jeśli mówimy o uczeniu z nauczycielem, to w trakcie uczenia wykorzystywany jest zbiór uczący zawierający przykładowe zadania wraz z ich wzorcowymi rozwiązaniami. W zbiorze uczącym zapisane są wartości zmiennych, które wprowadzane są na wejściu sieci oraz odpowiadające im wartości zmiennych wyjściowych, które mają charakter wzorców poprawnych odpowiedzi. Ten proces uczenia może być precyzyjnie kontrolowany, a tym samym może być prowadzony skutecznie [2, 11, 12, 13, 14, 15].

4. Elektrokardiogram

Elektrokardiogram (EKG) wraz z zapisem elektrycznej aktywności serca dostarcza wartościowych, dodatkowych informacji o czynności i budowie serca. EKG wychwytuje również aktywność innych mięśni, m. in. mięśni szkieletowych [16, 17, 18, 19, 20, 21].

Dopiero pod koniec XX wieku pojawiły się pierwsze próby automatycznej analizy sygnałów EKG. Pierwsze metody dotyczyły automatycznego wyznaczania prostych parametrów, takich jak np. czas trwania załamków itp. Pełna automatyczna analiza i diagnostyka sygnałów EKG ciągle stanowi wyzwanie badawcze. W tym zakresie próbuje się stosować wiele metod i algorytmów. Między innymi stosuje się metody sztucznej inteligencji czy rozpoznawania obrazów. Do rozpoznawania sygnałów EKG stosowane są różne rodzaje sieci neuronowych, od klasycznych sieci wielowarstwowych z nauczaniem metodą propagacji wstecznej błędu do wyrafinowanych sieci rekurencyjnych. W ostatnich latach ukazało się wiele publikacji, których przedmiotem jest analiza zapisu elektrokardiogramu pod kątem wykrycia zmian lub zaburzeń pracy układu krążenia, a przede wszystkim automatycznego rozpoznawania i klasyfikowania rytmów serca, co jest istotne z punktu widzenia diagnostyki medycznej. Do analizy w elektrokardiografii najczęściej stosuje się perceptron wielowarstwowy oraz sieć ze wsteczną propagacją. Sieci te przede wszystkim analizują rytm serca, jak również wykrywają zespół QRS oraz pozwalają na wczesne wykrycie arytmii.

5. Część badawcza

Sygnały elektrokardiograficzne wykorzystane w części eksperymentalnej zostały pozyskane z dwóch źródeł: ze zbiorów Gdańskiego Uniwersytetu Medycznego (GUM) oraz portalu internetowego

MIT-BIH Arrhythmia Database.

5.1. Dane pacjentów z GUM

Do przeprowadzenia części badawczej wykorzystane zostały wyniki badań elektrokardiograficznych Katedry Medycyny Rodzinnej Gdańskiego Uniwersytetu Medycznego. Ocenę diagnostyczna każdego przebiegu EKG przeprowadził doświadczony lekarz kardiolog. Każdy zapis EKG pochodził od innego pacjenta.

Dwóch pierwszych pacjentów doznało zawału ściany przednio-bocznej, czyli martwicy spowodowanej zatrzymaniem przepływu krwi przez główny pień gałęzi okalającej lewej tętnicy wieńcowej lub jej odnogę oraz odnogę boczną gałęzi zstępującej lewej tętnicy wieńcowej [22].

Trzeci pacjent cierpiał na zawał ściany dolnej, spowodowany okluzją dystalnego odcinka gałęzi okalającej lewej tętnicy wieńcowej lub tylno-bocznej odnogi gałęzi zstępującej prawej tętnicy wieńcowej [22].

Sygnały EKG, które zostały pozyskane do celów niniejszego projektu były zapisane w formacie PDF i XML. Niestety środowisko obliczeniowe Matlab nie obsługuje plików o powyższych rozszerzeniach. Fakt ten znacznie skomplikował pozyskiwanie danych. Należało więc otworzyć każdy plik za pomocą XML Notepad, skopiować potrzebne dane do Word, gdzie możliwa jest zamiana spacji na znak Enter (każda liczba zapisana została w oddzielnym wierszu), a następnie skopiować powstały plik do Notatnika. W ten sposób z 25 pojedynczych plików powstał jeden plik, który zawierał około 137 000 wierszy (danych wejściowych). Plik wyjściowy zawiera liczby 1 i 0 (które odpowiadają wartościom prawidłowy i fałszywy).

5.2. Dane pacjentów z MIT-BIH Arrhythmia Database

Kolejne 46 sygnałów elektrokardiograficznych pochodziło z ogólnodostępnej darmowej bazy sygnałów EKG: MIT-BIH Arryhythmia Database.

MIT-BIH dysponuje zbiorem ponad 4000 długoterminowych zapisów Holtera, które zostały uzyskane przez Beith Israel Hospital w latach 1975 do 1979. Około 60% z tych nagrań pochodziło od pacjentów hospitalizowanych. Baza danych zawiera 23 pozycje (ponumerowane od 100 do 124 włącznie, z niektórymi numerami zaginionymi) oraz 25 zapisów (ponumerowane od 200 do 234 włącznie, podobnie niektóre sygnały są zaginione).

Grupa pierwsza to reprezentacja różnych przebiegów i artefaktów arytmii. W drugiej grupie możemy się spotkać z komorowymi i nadkomorowymi zaburzeniami rytmu oraz zaburzeniami przewodzenia. Wśród grupy badanych było 25 mężczyzn w wieku od 32 do 89 lat oraz 22 kobiety w wieku od 23 do 89 lat [23].

Sygnały EKG, które zostały pozyskane z do celów niniejszego projektu były zapisane w formacie ATR, DAT i HEA. Plików tych nie da się bezpośrednio wczytać do Matlaba, do tego celu służy specjalnie stworzony skrypt, aby możliwe było odczytanie sygnałów i wyeksportowanie ich do pliku zmiennych Matlaba MAT. Ponadto, podobnie, jak w plikach pozyskanych GUM, utworzono pliki wyjściowe, które zawierają liczby 1 i 0 (odpowiadające wartościom prawidłowy i fałszywy).

6. Matlab

Jednym z celów projektu było zapoznanie się z oprogramowaniem Matlab oraz napisanie programu, który ma za zadnie analizować sygnały elektrokardiograficzne.

Matlab (MATrix LABoratory) to pakiet programowy, który umożliwia wykonywanie złożonych obliczeń numerycznych oraz wizualizację wyników. Jest to interaktywne środowisko zawierające funkcje obliczeniowe, graficzne oraz animacyjne. Daje również możliwość samodzielnego rozbudowania zakresu zastosowań poprzez tworzenie własnych skryptów i programów w dowolnym języku programowania [24, 25, 26, 27].

6.1. Symulacja i testowanie wybranych sieci neuronowych na przykładowych sygnałach EKG

Cały cykl programu w Matlabie można podzielić na pięć etapów. Zagadnienia te, a w szczególności proces tworzenia oraz uczenia sieci wielowarstwowych, zostały opisane dalej w tym podrozdziale.

6.2.1. Sieć wielowarstwowa I ze wsteczną propagacją błędów (z dwiema funkcjami aktywacji tangensoidalnymi i jedną liniową i liczbą iteracji 1000)

Etap I
Pierwszym krokiem jest utworzenie głównego interfejsu. Poniższy rysunek (rysunek 1) przedstawia wygląd głównego interfejsu.

Rys.1. Widok głównego okna – interfejs

Etap II
Drugim etapem jest utworzenie wszystkich pobocznych interfejsów, które uruchamiają się automatycznie po załadowaniu plików z danymi wejściowymi oraz wyjściowymi. Dane wejściowe to sygnały elektrokardiograficzne zapisane w formacie TXT. Pochodzą one z ogólnodostępnej bazy MIT-BIH. Pliki te uruchamiają się automatycznie. Na wejściu sieci jest podawanych 101 próbek danego sygnału. Liczba wejść sieci odpowiada liczbie badanych próbek sygnału.

Etap III
Etap trzeci polega na wybraniu odpowiedniej sieci oraz wprowadzeniu wszystkich parametrów, a następnie jej nauczaniu. Program daje do wyboru trzy rodzaje sieci: sieć liniową, perceptron i wielowarstwową ze wsteczną propagacja błędu. Wszystkie trzy rodzaje sieci zostały zaimplementowane w programie, chociaż ostatecznie, po wstępnym porównaniu ich właściwości, do rozpoznawania sygnałów EKG zastosowano jedynie klasyczną sieć wielowarstwową ze wsteczną propagacja błędu (p. rys. 2). Aby powstała sieć neuronowa, należy użyć jednej z trzech komend: *network*, *newff* lub *newpr*. Za pomocą *network* możliwe jest utworzenie własnych sieci neuronowych, które następnie są dostosowywane przez odpowiednie funkcje aktywacji: `newp`, `newlin` (sieć z liniową funkcją aktywacji), itp. Mianem funkcji aktywacji w sztucznej inteligencji określa się funkcję, według której obliczana jest wartość wyjścia neuronów sieci neuronowej. Komenda *newff* stosowana jest do tworzenia wielowarstwowej sieci neuronowej, której każda warstwa składa się z zadanej liczby neuronów o zadanych funkcjach aktywacji. Te funkcje aktywacji mogą być liniowe lub nieliniowe. *Newpr* (ang. *pattern recognition network*) to sieci, które mogą być przeznaczone do klasyfikowania wejścia według grup docelowych. Dane docelowe tej sieci powinny składać się z wektorów wszystkich wartości zerowych z wyjątkiem jednego elementu *i*, gdzie *i* jest klasą, którą reprezentują. Sieć utworzona w ten sposób nadaje się do dopasowywania obrazów oraz rozpoznawania wzorców.

Rys. 2. Wybór sieci

W niniejszej pracy opisano tworzenie sieci wielowarstwowej ze wsteczną propagacją błędów. W tym celu zastosowana została komenda *network*. Utworzona sieć miała jedno wejście oraz trzy warstwy; kolejno z 35, 18 i 1 neuronem w każdej warstwie. Każdą sieć może tworzyć dowolna liczba warstw. Następnie określono wartość funkcji przejścia warstw. Funkcje przejścia (aktywacji) neuronów z poszczególnych warstw sieci mogą być różne, dla każdej warstwy sieci funkcję przejścia określa jakiś parametr. W pierwszej i drugiej warstwie funkcja aktywacji warstwy to *tansig* (funkcja tangensoidalna), a w warstwie trzeciej występuje neuron z funkcją przejścia *purelin* (funkcja liniowa). Funkcja inicjalizacji określająca sieć to *initnw*. Następnie przedefiniowano wejściowe wagi sieci jako *rands*. Kolejna komenda służąca do tworzenia sieci neuronowej to *init(net)*, która inicjalizuje sieć. Następnie określono wartość funkcji oceny błędów odwzorowania sieci, w tym przypadku jest to *mse*, czyli błąd średniokwadratowy. MSE (ang. *Mean Squared Error*) jest wartością oczekiwaną kwadratu błędu, czyli różnicy pomiędzy estymatorem a wartością estymowaną.

Po wprowadzeniu wszystkich powyższych parametrów wybrano metodę uczenia. Omawiana sieć nauczona została metodą propagacji wstecznej Levenberg-Marquardta. Proces ten określono za pomocą komendy *trainlm*. Następnie wprowadzono parametry uczenia, wśród których można wyróżnić:

- *net.trainParam.lr* – współczynnik uczenia sieci,
- *net.trainParam.goal* – akceptowalny błąd,
- *net.trainParam.epochs* – liczba iteracji w czasie uczenia,
- *net.trainParam.show* – liczba kroków algorytmu uczenia, po którym wyświetlany jest komunikat.

Uczeniu zostało poddanych 39 sygnałów, w tym 19 nieprawidłowych i 20 prawidłowych. 10 sygnałów tworzyło zbiór testowy. Sygnały te zostały przygotowane jako zbiory dxx.txt oraz Txx.txt, gdzie xx to kolejny numer sygnału. Pierwszy zawiera próbki wybranego sygnału elektrokardiograficznego, a drugi odpowiednio wartości:

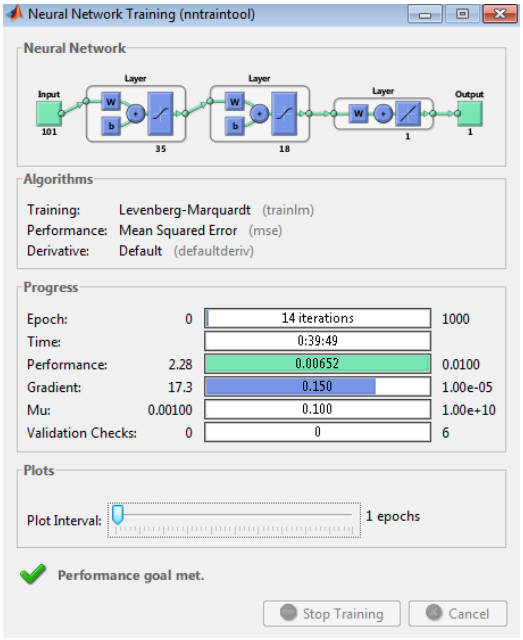
1 – sygnał prawidłowy (reprezentujący EKG zdrowego pacjenta),

0 – sygnał nieprawidłowy (reprezentujący EKG pacjenta z patologiami serca).

Jednakże sieć ta na wyjściu nie daje dokładnie wartości 0 i 1. Dlatego ustalono wartość progową wynoszącą 0,5. Sygnały prawidłowe są to te, dla których wartość na wyjściu jest wyższa od 0,5, a nieprawidłowe to te o wartości wyjściowej niższej od 0,5.

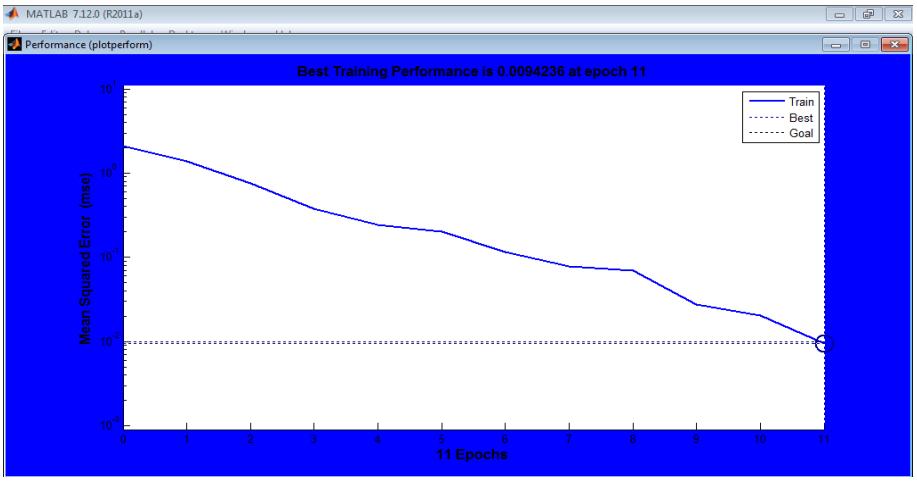
Etap IV

Proces nauczania takiej sieci w przybliżeniu trwał czterdzieści minut (p. rys. 3). Został on przeprowadzony na laptopie z procesorem Intel(R) Atom(TM) CPU N450 @1,67 GHz i pamięcią RAM 1,00 GB.



Rys. 3. Proces uczenia sieci

W etapie czwartym można było otrzymać wyniki. Z wykresu odczytano najlepsze osiągnięcie uczenia, które wynosi 0,0094236 i występuje w 11 epoce (p. rys. 4).



Rys.4. Wykres najlepszego wyniku

Następnie sprawdzano czy sieć nauczyła się klasyfikować prawidłowo 39 sygnałów uczących. Po podaniu na wejście sygnału nieprawidłowego symulacja powinna dać odpowiedź 0 oraz analogicznie dla sygnału prawidłowego odpowiedź 1. W tabeli 1 zaprezentowano zestaw wszystkich wyników symulacji. Symulacji dokonano za pomocą komendy: *sim(net,P(:,...)),* gdzie w miejsce trzech kropek wpisywano kolejne numery sygnału.

Tabela 1. Wyniki symulacji

Numer sygnału	Wartość symulacji	Numer sygnału	Wartość symulacji
1	-0,0109	20	0,9864
2	0,0282	21	0,9911
3	-0,0127	22	0,7589
4	-0,0132	23	1,0310
5	-0,0323	24	0,9815
6	-0,0226	25	0,9938
7	0,0547	26	0,9357
8	0,0076	27	0,9416
9	0,2998	28	0,9598
10	-0,0836	29	0,9780
11	-0,0538	30	0,9978
12	0,0325	31	1,0024
13	0,0138	32	0,8708
14	0,1150	33	1,0021
15	-0,0350	34	0,8456
16	0,0353	35	1,0135
17	0,0858	36	0,9502
18	0,1006	37	0,9825
19	0,0182	38	1,0062
		39	0,9721

Przed rozpoczęciem nauczania, należy dobrać odpowiednie parametry. To od nich będzie zależeć czas nauki oraz to czy sieć neuronowa nauczy się prawidłowo. Na podstawie powyższej tabelki można stwierdzić, iż sieć uczyła się prawidłowo.

Etap V

W ostatnim etapie sprawdzano czy sieć nauczyła się prawidłowo. W tym celu zastosowano te sygnały, które nie brały udziału w procesie uczenia. Wyniki sprawdzono analogicznie, jak podczas nauczania za pomocą komendy *sim(net,P(:,…))* (p. tab. 2).

Tabela 2. Wynik testu

Sygnał nieprawidłowy	Wynik testu
1	0,3505
2	0,4872
3	-1,5078
4	-0,6196
5	1,9224

Sygnał prawidłowy	Wynik testu
1	1,5032
2	1,6676
3	1,3846
4	1,3846
5	0,7156

Na podstawie powyższej tabeli można stwierdzić, iż sieć nie nauczyła się w 100% dobrze. Jednakże w 9 na 10 przypadków rozpoznała sygnały prawidłowo.

6.2.2. Sieć wielowarstwowa II ze wsteczną propagacją błędów (z trzema funkcjami aktywacji tangensoidalnymi i liczbą iteracji 15)

Etap I i II

Pierwsze dwa etapy – tworzenie głównego interfejsu oraz pobocznych interfejsów programu – były przeprowadzone identycznie, jak opisano poprzednio.

Etap III

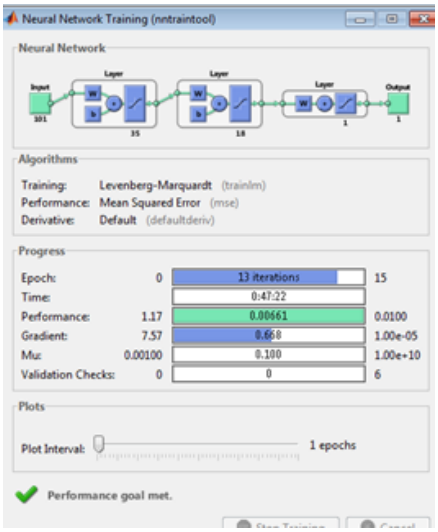
Etap trzeci to wybranie odpowiedniej sieci oraz wprowadzenie wszystkich parametrów. Podobnie jak w poprzednim przypadku, z trzech sieci wybrano sieć wielowarstwową ze wsteczną propagacją błędów. Została ona utworzona za pomocą komendy *network*. Sieć ta miała jedno wejście oraz trzy warstwy kolejno o 35, 18 i 1 neuronie w każdej warstwie. Każdą sieć może tworzyć dowolna liczba warstw. Następnie określono wartość funkcji przejścia warstw. W tym przypadku funkcje aktywacji wszystkich warstw były identyczne - *tansig* (funkcja tangensoidalna). Funkcja inicjalizująca tą sieć to *initnw*. Po wprowadzeniu tych parametrów została określona metoda uczenia – propagacja wsteczna Levenberg-Marquardta. Proces ten określono za pomocą komendy *trainlm*. Podobnie jak poprzednio, zadano parametry uczenia:

- *net.trainParam.lr* – współczynnik uczenia sieci,
- *net.trainParam.goal* – akceptowalny błąd,
- *net.trainParam.epochs* – liczba iteracji w czasie uczenia,
- *net.trainParam.show* – liczba kroków algorytmu uczenia, po którym wyświetlany jest komunikat.

Wszystkie te parametry były identyczne za wyjątkiem liczby iteracji w czasie uczenia, która została zmniejszona. Uczeniu zostało poddane 39 sygnałów, 19 nieprawidłowych i 20 prawidłowych. Pozostałe 10 sygnałów przeznaczono do testowania.

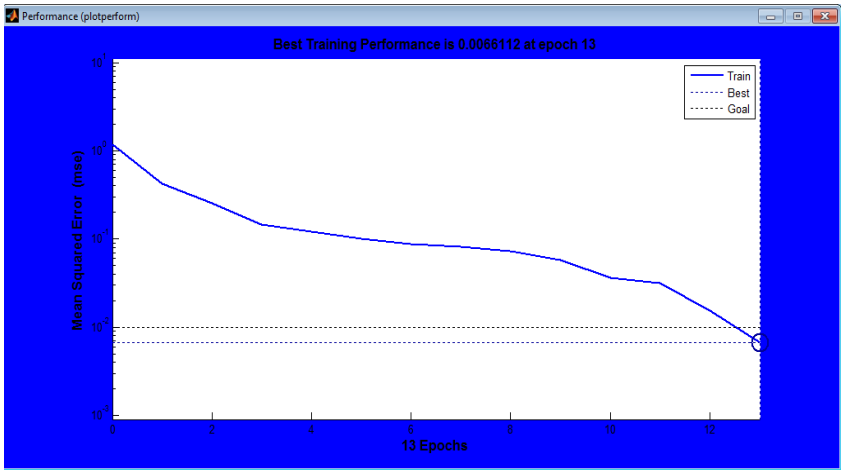
Etap IV

Proces nauczania takiej sieci w przybliżeniu trwał pięćdziesiąt minut (p. rys. 5).



Rys. 5. Proces uczenia sieci

W etapie czwartym można było zobaczyć wyniki. Z wykresu odczytano najlepsze osiągnięcie uczenia, które wynosi 0,0066112 i występuje w 13 epoce (p. rys. 6). Wykres ten przedstawia zależność błędu średniokwadratowego od epoki.



Rys.6. Wykres najlepszego wyniku

W tabeli 3 przedstawiono zestaw wszystkich wyników symulacji. Symulacji dokonano się za pomocą komendy: *sim(net,P(:,...)),* gdzie w miejsce trzech kropek wpisujemy numer sygnału.

Tabela 3. Wyniki symulacji

Numer sygnału	Wartość symulacji	Numer sygnału	Wartość symulacji
1	-0,0074	20	0,9732
2	0,0058	21	0,9588
3	-0,0133	22	0,8268
4	0,0406	23	0,9975
5	0,0088	24	0,9950
6	0,0017	25	0,9809
7	0,1578	26	0,9880
8	0,0163	27	0,9640
9	0,0966	28	0,9835
10	0,0357	29	0,9544
11	0,1122	30	0,9980
12	0,0950	31	0,9872
13	0,1691	32	0,8763
14	0,0860	33	0,9109
15	0,0332	34	0,8831
16	0,0782	35	0,9876
17	0,1992	36	0,9569
18	0,1973	37	0,9999
19	0,0162	38	0,9742
		39	0,9969

Etap V

W ostatnim etapie sprawdzano czy sieć nauczyła się prawidłowo. W tym celu wczytano wszystkie sygnały, które nie brały udziału w procesie uczenia. Wynik sprawdzano analogicznie jak podczas nauczania za pomocą komendy *sim(net,P(:,...))* (p. tab. 4).

Tabela 4a. Wyniki testu

Sygnał prawidłowy	Wynik testu
1	0,9990
2	0,6947
3	0,9999
4	0,9999
5	-0,0626

Tabela 4b. Wyniki testu

Sygnał nieprawidłowy	Wynik testu
1	0,4179
2	0,3115
3	0,1274
4	0,0942
5	0,9968

W tym przypadku sieć również nie rozpoznała w 100% procentach wszystkich sygnałów EKG na 10 sygnałów 8 sygnałów zostało rozpoznanych prawidłowo.

6.2.3. Sieć wielowarstwowa III ze wsteczną propagacją błędów (z dwoma funkcjami aktywacji tangensoidalnymi i jedną funkcją liniową symetryczną z nasyceniem oraz liczbą iteracji 15)

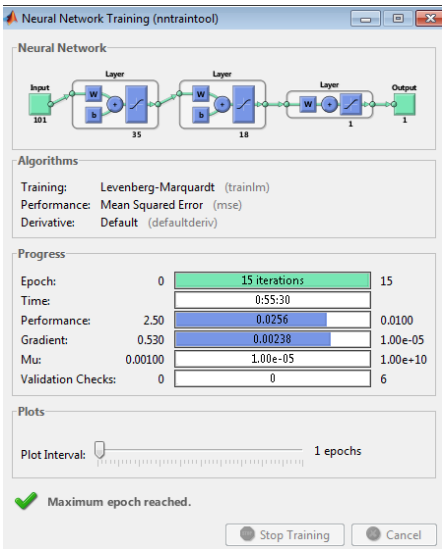
Etap I i II
Pierwsze dwa etapy – tworzenie głównego interfejsu oraz pobocznych interfejsów programu – jest takie samo, jak w poprzednich dwóch sieciach.

Etap III
Również w tym przypadku etap trzeci to wybranie odpowiedniej sieci oraz wybór wszystkich parametrów. Z trzech dostępnych sieci wybrano sieć wielowarstwową ze wsteczną propagacją błędów. Została ona utworzona za pomocą komendy *network*. Sieć ta miała jedno wejście oraz trzy warstwy kolejno o 35, 18 i 1 neuronie w każdej warstwie. Każdą sieć może tworzyć dowolna liczba warstw. Następnie określono wartości funkcji przejścia warstw. W tym przypadku funkcje aktywacji dla pierwszej i drugiej warstwy to *tansig* (funkcja tangensoidalna), a dla trzeciej *satlins* (funkcja liniowa symetryczna z nasyceniem). Podobnie jak w poprzednich dwóch sieciach funkcję inicjalizacji sieci określona została komendą *initnw*. Po wprowadzeniu tych parametrów została określona metoda uczenia – propagacja wsteczna Levenberg-Marquardta. Proces ten określono za pomocą komendy *trainlm*. Podobnie jak poprzednio, zadano odpowiednie parametry uczenia:

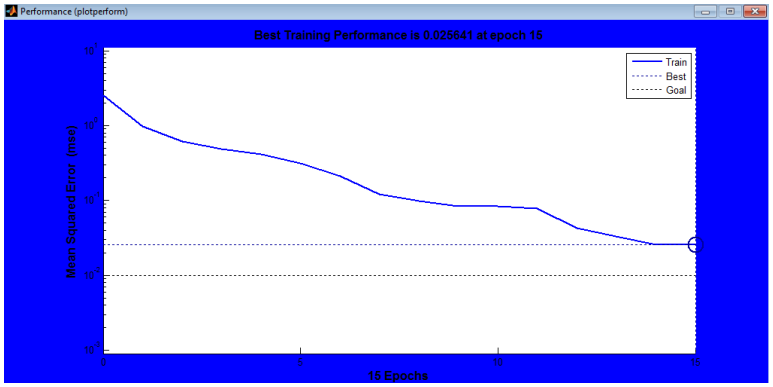
- *net.trainParam.lr* – współczynnik uczenia sieci,
- *net.trainParam.goal* – akceptowalny błąd,
- *net.trainParam.epochs* – liczba iteracji w czasie uczenia,
- *net.trainParam.show* – liczba kroków algorytmu uczenia, po którym wyświetlany jest komunikat.

Parametry uczenia są identyczne jak przy tworzeniu drugiej sieci. Uczeniu zostało poddane 39 sygnałów, 19 nieprawidłowych i 20 prawidłowych. Pozostałe 10 sygnałów przeznaczono do testowania.

Etap IV
Proces nauczania takiej sieci w przybliżeniu trwał pięćdziesiąt minut (p. rys. 7). W etapie czwartym możemy zobaczyć rezultaty – wyniki. Z wykresu możemy odczytać najlepsze osiągnięcie uczenia, które wynosi 0,025641 i występuje w 15 epoce (p. rys. 8). Wykres ten przedstawia zależność błędu średniokwadratowego od epoki.



Rys. 7. Proces uczenia sieci



Rys.8. Wykres najlepszego wyniku

W tabeli 4 przedstawiono zestaw wszystkich wyników symulacji. Symulacji dokonuje się za pomocą komendy: *sim(net,P(:,...)),* gdzie w miejsce trzech kropek wpisujemy numer sygnał.

Tabela 4. Wyniki symulacji

Numer sygnału	Wartość symulacji	Numer sygnału	Wartość symulacji
1	-1,0187e-004	20	1
2	-7,2645e-005	21	0,9998
3	3,0316e-004	22	0,9998
4	-4,5447e-004	23	1
5	1	24	1
6	1,2525e-004	25	1
7	0,0012	26	1
8	-2,3016e-004	27	1
9	-1,8081e-005	28	1
10	-0,0015	29	0,9992
11	1,1861e-004	30	1
12	9,9215e-004	31	1
13	1,5101e-004	32	1
14	0,0015	33	0,9984
15	9,3474e-004	34	1
16	0,0013	35	1
17	0,0010	36	1
18	0,0010	37	1
19	2,2527e-004	38	0,9995
		39	0,9993

Etap V

W ostatnim etapie przeprowadzono test na prawidłowe uczenie się sieci. Wczytano wszystkie sygnały, które nie brały udziału w procesie uczenia. Wynik sprawdzano analogicznie, jak podczas nauczania za pomocą komendy *sim(net,P(:,...))*.

Tabela 5. Wyniki testu

Sygnał prawidłowy	Wynik testu
1	0,3122
2	0,9820
3	-0,0745
4	-0,0745
5	-0,2801

Sygnał nieprawidłowy	Wynik testu
1	0,7246
2	0,0525
3	0,0340
4	0,1517
5	1

W tym przypadku sieć również nie rozpoznała w 100% procentach wszystkich sygnałów EKG. Na 10 sygnałów 4 sygnały zostało rozpoznanych prawidłowo.

6.2.4. Podsumowanie przeprowadzonych wyników

W tabeli 6 jest pokazane zestawienie wyników symulacji dla trzech sieci neuronowych zaprezentowanych w pracy. Na jej podstawie można wnioskować, iż najlepsza sieć to sieć wielowarstwowa II ze wsteczną propagacją błędów o funkcjach aktywacji tangensoidalnych.

Tabela 6. Zestawienie wszystkich wyników symulacji

Numer sygnału	Wartość symulacji		
	Sieć wielowarstwowa I	Sieć wielowarstwowa II	Sieć wielowarstwowa III
1	-0,0109	-0,0074	-1,0187e-004
2	0,0282	0,0058	-7,2645e-005
3	-0,0127	-0,0133	3,0316e-004
4	-0,0132	0,0406	-4,5447e-004
5	-0,0323	0,0088	1
6	-0,0226	0,0017	1,2525e-004
7	0,0547	0,1578	0,0012
8	0,0076	0,0163	-2,3016e-004
9	0,2998	0,0966	-1,8081e-005
10	-0,0836	0,0357	-0,0015
11	-0,0538	0,1122	1,1861e-004
12	0,0325	0,0950	9,9215e-004
13	0,0138	0,1691	1,5101e-004
14	0,1150	0,0860	0,0015
15	-0,0350	0,0332	9,3474e-004
16	0,0353	0,0782	0,0013
17	0,0858	0,1992	0,0010
18	0,1006	0,1973	0,0010
19	0,0182	0,0162	2,2527e-004
20	0,9864	0,9732	1
21	0,9911	0,9588	0,9998
22	0,7589	0,8268	0,9998
23	1,0310	0,9975	1
24	0,9815	0,9950	1
25	0,9938	0,9809	1
26	0,9357	0,9880	1
27	0,9416	0,9640	1
28	0,9598	0,9835	1

Numer sygnału	Wartość symulacji		
	Sieć wielowarstwowa I	Sieć wielowarstwowa II	Sieć wielowarstwowa III
29	0,9780	0,9544	0,9992
30	0,9978	0,9980	1
31	1,0024	0,9872	1
32	0,8708	0,8763	1
33	1,0021	0,9109	0,9984
34	0,8456	0,8831	1
35	1,0135	0,9876	1
36	0,9502	0,9569	1
37	0,9825	0,9999	1
38	1,0062	0,9742	0,9995
39	0,9721	0,9969	0,9993

Z kolei tabela 7 prezentuje wyniki przeprowadzonego testu. Tutaj również najlepiej test przeszła sieć wielowarstwowa II ze wsteczną propagacją błędów o dwóch funkcjach aktywacji tangensoidalnych i jednej liniowej, która na 10 zadanych sygnałów 9 rozpoznała prawidłowo.

Tabela 7. Zestawienie wszystkich wyników testu

	Numer sygnału	Wynik testu		
		Sieć wielowarstwowa I	Sieć wielowarstwowa II	Sieć wielowarstwowa III
Sygnał prawidłowy	1	1,5032	0,9990	0,3122
	2	1,6676	0,6947	0,9820
	3	1,3846	0,9999	-0,0745
	4	1,3846	0,9999	-0,0745
	5	0,7156	-0,0626	-0,2801
Sygnał nieprawidłowy	1	0,3505	0,4179	0,7246
	2	0,4872	0,3115	0,0525
	3	-1,5078	0,1274	0,0340
	4	-0,6196	0,0942	0,1517
	5	1,9224	0,9968	1

7. Wnioski

Ważnym elementem była próba wykorzystania środowiska Matlab do stworzenia prostego programu realizującego zadanie rozpoznawania i rozróżniania pomiędzy prawidłowymi i nieprawidłowymi sygnałami EKG. Do analizowania sygnałów EKG wybrano wielowymiarową jednokierunkową sieć ze wsteczna propagacją błędu. Utrudniony dostęp do szerokiej klasy sygnałów EKG w postaci próbkowanej oraz bardzo szeroki i skomplikowany zakres badań nie pozwoliły na uzyskanie skończonych i pełnych wyników. Temat ten wymaga jeszcze wielu dodatkowych badań, rozważenia większej klasy sieci neuronowych, większej liczby parametrów, jak też dostępu do większego zbioru próbkowanych sygnałów EKG. Badania te pokazały, że podejście w którym kolejne próbki badanego sygnału EKG podajemy na kolejne wejścia sieci może się okazać interesujące. Przeprowadzone badania i symulacje nasunęły wnioski, że zmieniając tylko funkcję aktywacji w ostatniej warstwie oraz liczbę iteracji, najlepsze wyniki osiąga sieć o dwóch funkcjach aktywacji typu tangensoidalnego i jednej liniowej. W tym przypadku dla 10 sygnałów testowych sieć rozpoznała prawidłowo aż 9 z nich. Zmieniając krok po kroku wybrane parametry można zapewne osiągnąć jeszcze lepsze wyniki.

LITERATURA

[1] W. Duch: *Fascynujący świat komputerów, Sztuczna inteligencja*, Poznań 1997.

[2] R. Tadeusiewicz, G. Paliwoda-Pękosz, P. Lula: *Metody sztucznej inteligencji i ich zastosowania w ekonomii i zarządzaniu*, Akademia Ekonomiczna w Krakowie, Kraków 2003.

[3] H. Kwaśnicka: *Sztuczna Inteligencja i Systemy Ekspertowe Rozwój*, Perspektywy; Katedra Inżynierii Oprogramowania, Wyższa Szkoła Zarządzania i Finansów, Wrocław 2005

[4] R. Tadeusiewicz: *Wykłady na temat sieci neuronowych*, Laboratorium Biocybernetyki, Katedra Automatyki na AGH
[5] http://www.neurolotto.com/index.php?c=37&padd=1
[6] M. Nałęcza: *Biocybernetyka i Inżynieria Biomedyczna 2000 tom 6 Sieci neuronowe*, Akademicka Oficyna Wydawnicza EXIT, Warszawa 2000.
[7] R. Tadeusiewicz: *Elementarne wprowadzenie do techniki sieci neuronowych z przykładowymi programami*, Akademicka Oficyna Wydawnicza PLJ, Warszawa 1998.
[8] R. Tadeusiewicz: *Wprowadzenie do sieci neuronowych*, StatSoft, Kraków 2001.
[9] Ł. Sanocki: *Wstęp do sieci neuronowych*, Katedra Elektroniki, Akademia Górniczo-Hutnicza, Kraków, 2005.
[10] Sieci neuronowe, Katedra Inżynierii Komputerowej, Politechnika Częstochowska, Częstochowa.
[11] J. Korbicz, A. Obuchowicz, D. Uciński: *Sztuczne Sieci Neuronowe. Podstawy i Zastosowania*, Akademicka Oficyna Wydawnicza PLJ, Warszawa 1994.
[12] A. Duraj: *Algorytmy rozpoznawania zespołów QRS w sygnałach elektrokardiograficznych pochodzących od pacjentów z wszczepionym układem stymulującym* – rozprawa doktorska, Uniwersytet Zielonogórski, Wydział Elektroniki Informatyki i Telekomunikacji, Zielona Góra 2007.
[13] H. Kwaśnicka: *Obliczenia Ewolucyjne w sztucznej inteligencji*, Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 1999.
[14] S. Osowski: *Sieci neuronowe w ujęciu algorytmicznym*, Wydawnictwo Naukowo-Techniczne, Warszawa 1996.
[15] http://kopernik.matfiz.polsl.pl/www/marcin.wozniak/inne/sieci_neuronowe/index.html
[16] T. Garcia, N. Holtz: *EKG Sztuka interpretacji*, MediPage, Warszawa 2007.
[17] A. Houghton, D. Gray: *EKG – jasno i zrozumiale*, α-medica, 2008, s. 10–30.
[18] T. Tomasik, A. Windak, A. Skalska, J. Kulczycka-Życzkowska, A. Kocemba: *Elektrokardiografia dla lekarza praktyka. Podręcznik z materiałami samokształceniowymi 195 pytań testowych, 70 ćwiczeń praktycznych*, Uniwersyteckie Wydawnictwo Medyczne Vesalius, Kraków 1998.
[19] G. Cliffird, L. Tarassenko, N. Townsend: *Fusing Conventional ECG QRS Detection Algorithms with an Auto-associative Neural Network for the Detection of Ectopic Beats*, International Conference, Beijing 2000.
[20] S. Yang, G. Yang: *ECG Pattern Recognition Based on Wavelet Transform and BP Neural Network*, Jinggangshan, China, 2–4 April 2010, s. 246–249.
[21] B. Dąbrowska, A. Dąbrowski: *Podręcznik elektrokardiografii*, Wydawnictwo Lekarskie PZWL, Warszawa, 1993.
[22 http://csk.umed.lodz.pl/nefrologia/ekg%202.pdf
[23] http://www.physionet.org/physiobank/database/mitdb/
[24] R. Cegieła, A. Zalewski: *Matlab – obliczenia numeryczne i ich zastosowania*, Wydawnictwo Nakom, Poznań 1997.
[25] T. Kucharski: *Programowanie obliczeń inżynierskich*, Wydawnictwo Politechniki Gdańskiej, Gdańsk 2000.
[26] P. Rudra: *Matlab 7 dla naukowców i inżynierów*, PWN, Warszawa 2007.
[27] B. Mrozek, Z. Mrozek: *Matlab uniwersalne środowisko do obliczeń naukowo-technicznych*, Wydawnictwo PLJ, Warszawa 1996.

otrzymano / submitted: 08.05.2014
wersja poprawiona / revised version: 13.05.2014
zaakceptowano / accepted: 30.06.2014