

Politechnika Wrocławska
Wydział Elektroniki

Kierunek: Automatyka i Robotyka (AIR)
Specjalność: Komputerowe Systemy Zarządzania Procesami
Produkcyjnymi (ARS)

PRACA DYPLOMOWA
(MAGISTERSKA)

**Zastosowanie sieci neuronowych do
sterowania bio-manipulatorem**

Autor:
Sebastian Busz

Prowadzący pracę:
dr inż. Andrzej Wolczowski

/pieczęć jednostki
prowadzącej specjalność/

Ocena pracy:
/stopień, słownie, podpis odręczny/

Wrocław 2003

Spis treści

SPIS TREŚCI	- 2 -
1. WPROWADZENIE	- 4 -
2. ZAKRES PRACY	- 6 -
3. BIOPROTEZA	- 7 -
3.1. RYS HISTORYCZNY	- 7 -
3.2. IDEA ZRĘCZNEJ BIOPROTEZY	- 8 -
4. ROZPOZNAWANIE OBIEKTÓW.....	- 10 -
4.1. WPROWADZENIE	- 10 -
4.2. POMIAR SYGNAŁU	- 12 -
4.2.1. <i>Natura sygnałów miopotencjałów.....</i>	<i>- 12 -</i>
4.2.2. <i>Pomiar miopotencjałów.....</i>	<i>- 12 -</i>
4.2.3. <i>Projekt wzmacniacza pomiarowego.....</i>	<i>- 14 -</i>
4.2.4. <i>Parametry układu pomiarowego.....</i>	<i>- 15 -</i>
4.2.5. <i>Konwersja na postać cyfrową.....</i>	<i>- 17 -</i>
4.3. EKSTRAKCJA CECH	- 18 -
4.3.1. <i>Transformata Fouriera</i>	<i>- 18 -</i>
4.3.2. <i>Inne sposoby ekstrakcji cech.....</i>	<i>- 20 -</i>
4.4. REDUKCJA WYMIARU WEKTORA CECH	- 20 -
4.5. KLASYFIKACJA	- 22 -
4.5.1. <i>Pojęcia podstawowe.....</i>	<i>- 22 -</i>
4.5.2. <i>Klasyfikator Bayesowski</i>	<i>- 23 -</i>
4.5.3. <i>Klasyfikacja ze zbiorem uczącym.....</i>	<i>- 25 -</i>
4.5.4. <i>Ocena jakości rozpoznawania</i>	<i>- 26 -</i>
5. WIELOWARSTWOWA SIEĆ JEDNOKIERUNKOWA.....	- 28 -
5.1. WSTĘP	- 28 -
5.2. NEURON	- 28 -
5.3. STRUKTURA SIECI MLP	- 31 -
5.4. UCZENIE SIECI – ALGORYTM WSTECZNEJ PROPAGACJI BŁĘDÓW	- 32 -
5.5. SIEĆ WIELOWARSTWOWA JAKO KLASYFIKATOR	- 35 -
5.6. MOMENTUM	- 36 -
5.7. DOBÓR WSPÓŁCZYNNIKA UCZENIA.....	- 36 -
5.7.1. <i>Stały współczynnik uczenia – MLP1</i>	<i>- 37 -</i>
5.7.2. <i>Dobór współczynnika na podstawie wyników klasyfikacji – MLP2</i>	<i>- 37 -</i>
5.7.3. <i>Adaptacyjny dobór współczynnika uczenia – MLP3.....</i>	<i>- 37 -</i>
5.7.4. <i>Inne metody doboru współczynnika uczenia</i>	<i>- 38 -</i>
5.8. INNE METODY UCZENIA SIECI WIELOWARSTWOWYCH	- 38 -
6. SIEĆ LVQ.....	- 39 -
6.1. WSTĘP	- 39 -
6.2. SIECI SAMOORGANIZUJĄCE KOHONENA	- 39 -
6.3. KLASYFIKATOR LVQ	- 41 -
6.4. METODY UCZENIA	- 41 -
6.4.1. <i>LVQ1.....</i>	<i>- 41 -</i>
6.4.2. <i>LVQ2.....</i>	<i>- 42 -</i>
6.4.3. <i>LVQ3.....</i>	<i>- 43 -</i>
6.4.4. <i>Modyfikacje.....</i>	<i>- 43 -</i>
6.5. UWAGI DOTYCZĄCE UCZENIA.....	- 44 -
6.5.1. <i>Inicjalizacja wag początkowych.....</i>	<i>- 44 -</i>
6.5.2. <i>Przebieg uczenia</i>	<i>- 44 -</i>
6.5.3. <i>Problem nieużywanych neuronów.....</i>	<i>- 45 -</i>
7. WYNIKI EKSPERYMENTÓW	- 46 -
7.1. WYRÓŻNIONE KLASY	- 46 -
7.2. POMIAR SYGNAŁÓW	- 47 -

7.2.1.	<i>Ułożenie elektrod</i>	- 47 -
7.2.2.	<i>Zamiana na postać cyfrową</i>	- 47 -
7.2.3.	<i>Sposób tworzenia bazy pomiarów</i>	- 48 -
7.3.	ZBIÓR UCZĄCY I TESTUJĄCY	- 49 -
7.4.	SIECI MLP	- 49 -
7.4.1.	<i>Dobór struktury sieci</i>	- 49 -
7.4.2.	<i>Dobór współczynnika uczenia i momentum dla MLP1</i>	- 50 -
7.4.3.	<i>Dobór parametrów dla metody MLP2</i>	- 53 -
7.4.4.	<i>Dobór parametrów dla metody MLP3</i>	- 55 -
7.5.	SIECI LVQ	- 57 -
7.6.	PORÓWNANIE METOD KLASYFIKACJI	- 63 -
7.7.	PODSUMOWANIE	- 64 -
LITERATURA		- 65 -
DODATEK A – OPIS PROGRAMU ARTIFICIALHAND		- 66 -
DODATEK B - OPIS PROGRAMU EMGRECORDER		- 67 -
DODATEK C – IMPLEMENTACJA WYBRANYCH METOD		- 68 -

1. Wprowadzenie

Trudno jest przecenić rolę sprawnych rąk i dłoni w życiu człowieka. Większość codziennych zadań wykonujemy używając bardzo sprawnych kończyn górnych. Zdarza się jednak, że człowiek jest pozbawiony części kończyny z powodu wypadku lub choroby. Przekonuje się wtedy jak bardzo komplikuje to najprostsze nawet czynności. Z problemem tym ludzkość spotyka się już od długiego czasu. Wykształcono wiele sposobów nadrabiania braku dłoni lub ręki poprzez zastosowanie różnego rodzaju protez, różniących się sposobem wykonania i funkcjonalnością. Najprostszą z nich jest hak, stosowany nawet do dziś. Kolejnym etapem było stworzenie protez jak najbardziej przypominających wyglądem prawdziwą kończynę. Obecnie wykorzystywane materiały sprawiają, że powierzchnia protezy wygląda jak skóra, kształt protezy odpowiada kształtowi brakującej ręki, nawet paznokcie wyglądają autentycznie. Jednocześnie starano się zapewnić odpowiednią funkcjonalność, na początku przez mechaniczne zamykanie i otwieranie protezy poprzez system linek, a później poprzez napęd elektryczny. W przypadku protez napędzanych silnikami elektrycznymi, ważną rolę zaczął odgrywać sposób sterowania napędami, a konkretnie źródło informacji o tym, jak proteza powinna się zachować w danej chwili.

Najbardziej naturalnym podejściem jest wykorzystanie do sterowania bioprotezą dłoni sygnałów elektrycznych z mięśni pozostających w kikucie. Pomiar i rejestracja sygnału aktywności elektrycznej mięśni nazywany jest elektromiografią (EMG). Do pomiaru EMG konieczne jest zastosowanie odpowiednio skonstruowanego toru pomiarowego składającego się z elektrod zapewniających dobry styk z powierzchnią skóry oraz różnicowych wzmacniaczy pomiarowych. Pomiar różnicowy jest konieczny dlatego, że sygnał EMG ma niewielką amplitudę, nawet o kilka rzędów mniejszą niż zakłócenia pochodzące z otoczenia. Sygnał mierzy się w dwóch punktach, a następnie odejmuje od siebie. W ten sposób część wspólna sygnałów zawierająca zakłócenia jest eliminowana, natomiast różnica powodowana aktywnością mięśni jest wzmacniana. Parametry sygnału zmierzonego w ten sposób na przedramieniu można wykorzystać do sterowania bioprotezą dłoni.

Najprostszym sposobem sterowania jest wykorzystanie wartości skutecznej zmierzonego sygnału. Sterownik może działać w ten sposób, że zamyka protezę po osiągnięciu przez wartość skuteczną odpowiedniego poziomu, co powodowane jest napięciem mięśni. Otwieranie protezy może następować po zaniku aktywności mięśni lub przy odpowiednim poziomie wartości skutecznej sygnału mierzonego dla grupy mięśni antagonistycznych, co wymaga jednak zastosowania dwóch kanałów pomiarowych.

Gdy rozwój technologii umożliwił wykonanie lekkich protez o wielu stopniach swobody, tzn. takich, które mają wiele przegubów, a przez to większe możliwości ruchu, pojawił się problem skutecznego sterowania takimi protezami na podstawie sygnałów EMG. Sterowanie przy wykorzystaniu tylko wartości skutecznej okazało się niewystarczające, nawet przy wielu kanałach pomiarowych. Zaczęto prowadzić badania nad wykorzystaniem innych parametrów sygnału EMG, takich jak widmo częstotliwościowe. Badania wykazały, że zawartość składowych częstotliwościowych sygnału mierzonego powierzchniowo zmienia się w zależności od tego, które grupy mięśni są aktywne, a zatem od tego, jaki ruch jest wykonywany lub w jakim stanie znajduje się dłoń. Analiza widma częstotliwościowego sygnału EMG umożliwia rozpoznanie stanu dłoni, co może być wykorzystane przez sterownik bioprotezy do odpowiedniego sterowania przegubami.

Ogólnie, problem pomiaru, wydobywania z pomiarów pewnych cech i określenia klasy, do której zalicza się obiekt opisany tymi cechami znany jest jako problem rozpoznawania. Ostatni etap procesu rozpoznawania, czyli przypisanie klasy do obiektu reprezentowanego przez wektor cechy, nazywany jest klasyfikacją.

W przypadku rozpoznawania stanów dłoni na podstawie sygnałów EMG proces przebiega następująco

- za pomocą elektrod i różnicowych wzmacniaczy pomiarowych mierzony jest sygnał aktywności mięśni
- obliczana jest transformata Fouriera sygnału, na podstawie której wyznacza się widmo częstotliwościowe
- z widma wydobywa się wektor cech charakteryzujący stan dłoni
- klasyfikator określa numer klasy, do której należy wektor cech, przy czym numery klas odpowiadają wyróżnionym stanom dłoni

W całym procesie rozpoznawania kluczową rolę odgrywa klasyfikacja. Istnieją dwa główne podejścia do tego problemu mające różne inspiracje. Są to klasyfikacja statystyczna i neuronowa. W obu przypadkach wymagany jest tzw. zbiór uczący, tj. zbiór wektorów cech wraz z numerami klas, do których należą obiekty opisane tymi cechami. Przy klasyfikacji statystycznej zbiór uczący wykorzystywany jest do wyznaczenia statystycznych parametrów poszczególnych klas, co umożliwia stworzenie matematycznego modelu badanego zjawiska, na podstawie którego dokonuje się klasyfikacji. Klasyfikatory neuronowe próbują naśladować działanie ludzkiego mózgu, poprzez modelowanie struktury połączeń dużej liczby elementarnych jednostek obliczeniowych, jakimi są neurony. W tym przypadku zbiór uczący wykorzystywany jest do nauczania sieci neuronowej prawidłowej reakcji na podany na wejście wektor cech, tj. zaliczenie tego wektora do odpowiedniej klasy.

Ze względu na skuteczność rozpoznawania, na szczególną uwagę zasługuje druga grupa klasyfikatorów, wśród których wyróżnić należy klasyfikator oparty na wielowarstwowej jednokierunkowej sieci typu perceptronowego – MLP (z ang. Multi Layer Perceptron) oraz sieci LVQ (z ang. Learning Vector Quantization).

Sieci MLP są już klasycznym rozwiązaniem, służącym do odwzorowywania nawet bardzo złożonych funkcji. Sieć taka składa się z kilku warstw połączonych ze sobą neuronów. Neurony modelowane są przez układ sumatora ważonego połączonego z blokiem sigmoidalnej funkcji aktywacji. Uczenie sieci MLP polega na doborze wartości wag neuronów w taki sposób, aby zminimalizować błąd rozpoznawania przykładów ze zbioru uczącego. Do uczenia można zastosować szereg algorytmów, z których najbardziej znany, to algorytm najszybszego spadku wykorzystujący do obliczania gradientu metodę wstecznej propagacji błędów.

Sieci LVQ zawierają tylko jedną warstwę neuronów, z których każdy przypisany jest do jednej z klas. Wagi neuronów odpowiadają cechom obiektu należącego do danej klasy. Klasyfikacja polega na znalezieniu neuronu o wagach najbardziej podobnych (najbliższych w sensie przyjętej metryki) do rozpoznawanego wektora cech i przyjęciu za wynik numeru klasy, do której należy zwycięski neuron. Do uczenia sieci LVQ można zastosować jedną z metod zaproponowanych przez Kohonena.

Jak wcześniej wspomniano, wynik klasyfikacji może stanowić podstawę decyzji o działaniu, jaką podejmuje sterownik bioprotezy. Możliwe jest wykorzystanie dodatkowych źródeł informacji o stanie dłoni i otoczenia, takich jak czujniki zbliżeniowe zamontowane wewnątrz dłoni, czujniki siły uścisku montowane w palcach, czy czujniki poślizgu. Sterownik, korzystając ze wszystkich zebranych informacji, byłby w stanie reagować na różne sytuacje w sposób zbliżony do podświadomych reakcji człowieka, np. mocniejsze chwycenie wyślizgującego się przedmiotu. Jednak podstawowe funkcje powinny być sterowane świadomie, a najbardziej naturalnym sposobem wydawania komend bioprotezie jest sterowanie z wykorzystaniem odruchów napinania mięśni przy wykonywaniu różnych czynności. Od budowy systemu rozpoznającego zależy, jak wiele informacji można wydobyć z napinanych mięśni.

2. Zakres pracy

Celem pracy było zbadanie możliwości wykorzystaniem sieci neuronowych do sterowania bioprotezą na podstawie sygnałów bioelektrycznych mięśni mierzonych na przedramieniu człowieka. Ze względu na ilość poruszanych zagadnień ograniczono się do etapu rozpoznawania stanu dłoni na podstawie miopotencjałów, nie zajmowano się wykorzystaniem wyniku klasyfikacji do sterowania przegubami protezy. Zakres pracy obejmuje wszystkie etapy rozpoznawania, od pomiaru aktywności elektrycznej mięśni do klasyfikacji neuronowej.

Część pracy związana z projektem i wykonaniem układu pomiarowego oraz rejestracją sygnałów miopotencjałów zrealizowana została przy współpracy z Tomaszem Białobrzegiem. W swojej pracy ([14]) kolega Białobrzeg zajmuje się bardziej szczegółowo problemami pomiaru i przetwarzania na postać cyfrową oraz analizą miopotencjałów. W tej pracy główny nacisk kładzie się na zagadnienia związane z rozpoznawaniem, a w szczególności na klasyfikację z użyciem sieci neuronowych.

Rozdział 3 pracy zawiera krótkie omówienie rozwoju protez ręki. Zaprezentowana jest idea współczesnej bioprotezy sterowanej miopotencjałami.

Rozdział 4 wprowadza we wszystkie etapy rozpoznawania, poczynając od pomiaru miopotencjałów, przez ekstrakcję cech, po klasyfikację. Omówione są właściwości sygnałów elektrycznych mięśni. Przedstawiony jest projekt układu pomiarowego wykonany wspólnie z kolegą Białobrzegiem. Opisane są zagadnienia związane z ekstrakcją cech i klasyfikacją, zarówno statystyczną, jak i neuronową.

Rozdział 5 zawiera omówienie wielowarstwowej sieci neuronowej z sigmoidalną funkcją aktywacji stosowanej jako klasyfikator. Opisana jest budowa i zasada działania sieci oraz sposoby uczenia, ze szczególnym uwzględnieniem wstecznej propagacji błędów. Przedstawione są sposoby dobierania współczynnika uczenia, będącego podstawowym parametrem wpływającym na proces nauki.

Rozdział 6 poświęcony jest sieciom LVQ. Zaprezentowano ideę budowy i działania sieci samoorganizujących Kohonena, a następnie sieci LVQ, stanowiących odmianę sieci samoorganizujących. Przedstawiono podstawowe algorytmy uczenia sieci.

Rozdział 7 przedstawia wyniki pomiarów i badań, wraz ze szczegółowym omówieniem sposobu przeprowadzania kolejnych eksperymentów.

Rozdział 8 stanowi podsumowanie pracy zawierające wnioski końcowe i wskazujące sugerowane kierunki dalszych badań związanych z tą dziedziną.

Dodatek A poświęcony jest opisowi programu *ArtificialHand*, który został napisany na potrzeby pracy. Program ten umożliwia rejestrowanie i analizę zmierzonych miopotencjałów, klasyfikację za pomocą metod statystycznych i neuronowych oraz wizualizację wyniku klasyfikacji w postaci symulacji bioprotezy.

Dodatek B zawiera opis programu *EMGRecorder*, który używany był do stworzenia bazy sygnałów, na podstawie których przeprowadzane były wszystkie badania.

Dodatek C zawiera kod wybranych fragmentów programu.

3. Bioproteza

3.1. Rys historyczny

Protezą nazywa się mechaniczny przyrząd lub sztucznie stworzony aparat, którego zadaniem jest zastępowanie części ciała lub maskowanie jej braku. Ogólnie protezy można podzielić na dwie grupy: protezy kosmetyczne i kinetyczne. Protezy kosmetyczne mają za zadanie jedynie estetyczne maskowanie ubytku ciała, np. sztuczna pierś lub szklane oko. Zadaniem protez kinetycznych jest zastąpienie funkcji wykonywanych przez brakujący organ ciała. Do protez zalicza się także takie aparaty jak sztuczna nerka, czy sztuczne płuco-serce. Najpopularniejszym rodzajem protez są bez wątpienia protezy dentystyczne.

Ze względu na złożoność i różnorodność wykonywanych funkcji na szczególną uwagę zasługują protezy kończyn górnych. Pierwsze protezy tego typu miały formę metalowego haka i umożliwiały osobie pozbawionej dłoni wykonywanie podstawowych czynności. W XX wieku, dzięki wykorzystaniu metali lekkich i tworzyw sztucznych, protezy stały się lekkie, higieniczne i łatwe w użyciu. Zaczęły bardziej przypominać brakującą kończynę. Obecnie duży nacisk kładzie się na dwa aspekty: naturalny wygląd oraz dużą funkcjonalność.

Wiele firm na świecie (np. Otto Bock, Scandinavian Orthopaedic Laboratory) oferuje swoim pacjentom protezy wykonywane na indywidualne zamówienie, które niezwykle wiernie odtwarzają wygląd sprawnej kończyny. Ma to szczególnie duże znaczenie w przypadku protez rąk i dłoni, które są najbardziej widoczne w codziennym życiu. Do budowy protez używa się silikonów oddających strukturę skóry pacjenta. Odcień koloru protezy dobierany jest indywidualnie. Ponadto dużą wagę przywiązuje się do wykończenia szczegółów takich jak kształt paznokci, czy nawet linie papilarne. W efekcie trudno odróżnić protezę od naturalnej części ciała. Na Rys. 3.1.1 przedstawiono protezę firmy Otto Bock ([15]), w której skład wchodzi mechaniczny chwytak oraz silikonowa rękawiczka kosmetyczna wykonywana zgodnie z indywidualnymi potrzebami pacjenta.



**Rys. 3.1.1. Chwytak mechaniczny protezy firmy Otto Bock
wraz z dwoma rękawiczkami kosmetycznymi**

Drugi kierunek rozwoju dotyczy zwiększenia funkcjonalności protez. Pierwsze próby poprawy funkcjonalności polegały na zastosowaniu systemu linek łączących protezę ze zdrowymi częściami ciała w taki sposób, aby przy odpowiednim ruchu np. ramion proteza samoczynnie się zamykała bądź otwierała. Rozwiązania takie stosowane są do dziś, zarówno z protezami typu hak, jak i protezami silikonowymi (patrz Rys. 3.1.1).

Kolejnym krokiem było zastosowanie napędu elektrycznego do zamykania i otwierania protezy. Jednocześnie zaczęto prowadzić prace nad sposobem sterowania protezą. Najbardziej naturalne jest sterowanie wykorzystujące sygnały elektryczne z mięśni

pozostających w przedramieniu (tzw. miopotencjały). Należy zauważyć, że nawet po amputacji pourazowej zazwyczaj część mięśni przedramienia pozostaje nienaruszona i pacjent ma możliwość ich napinania. W przypadku dzieci z wadami wrodzonymi stwierdzono, że bardzo szybko uczą się one odruchów napinania (tzw. stereotypów ruchu). Dlatego badania nad wykorzystaniem aktywności elektrycznej mięśni do sterowania protezą są jak najbardziej uzasadnione i od wielu lat prowadzone przez wiele ośrodków badawczych.

Najprostszym sposobem wykorzystania miopotencjałów do sterowania protezą jest pomiar wartości skutecznej sygnału elektrycznego mięśni i zamykanie (bądź otwieranie) protezy w przypadku, gdy wartość ta przekracza pewien ustalony próg. Działanie odwrotne protezy, tzn. otwieranie (bądź zamykanie) może następować przy braku odpowiedniej aktywności mięśni. W przypadku zastosowania dwóch kanałów pomiarowych, działanie odwrotne może następować, gdy aktywność mięśni antagonistycznych osiągnie ustalony próg. Protezy wykorzystujące ten typ sterowania, produkowane są m.in. przez firmę Otto Bock.

Rozwinięciem przedstawionej powyżej idei jest sterowanie z różnymi wartościami prędkości otwierania, zamykania oraz siły ścisku, w zależności od poziomu aktywności elektrycznej mięśni. Prace nad tego typu protezami prowadzone są także na Politechnice Wrocławskiej – patrz [16].

Pod koniec XX wieku pojawiły się technologiczne możliwości budowy lekkich protez o większej liczbie przegubów, które charakteryzują się znacznie szerszym zakresem możliwych do wykonania ruchów, w porównaniu do prostych protez typu otwórz-zamknij. Protezy tego typu mogą przywracać podstawowe funkcje dłoni, jednak problemem jest odpowiednie sterowanie umożliwiające sprawne i pewne wykonywanie określonych ruchów. Wykorzystanie jedynie informacji o wartości skutecznej sygnału elektrycznego mięśni jest niewystarczające, nawet przy wykorzystaniu wielu kanałów pomiarowych. Do rozwiązania tego problemu potrzebne jest inne, znacznie bardziej złożone podejście, wykorzystujące współczesne techniki rozpoznawania.

3.2. Idea zręcznej bioprotezy

Wiele ośrodków badawczych na świecie (np. japoński uniwersytet Hakkaido – patrz [17]) prowadzi intensywne badania nad budową i sterowaniem protez wiernie naśladujących funkcje dłoni. Proteza taka wyposażona jest w pięć palców, z których każdy może być sterowany indywidualnie. Kształt i waga sztucznej dłoni mają być zbliżone do kształtu i wagi jej rzeczywistego odpowiednika. Budowa protezy umożliwia wykonywanie bardzo złożonych i precyzyjnych ruchów, pod warunkiem, że zapewnione zostanie odpowiednie sterowanie.

Jak przedstawiono w poprzednim punkcie, sterowanie za pomocą miopotencjałów pojedynczą funkcją typu otwórz-zamknij nie jest trudnym problemem. Od lat dostępne są na rynku protezy sterowane w ten sposób. Jednak w przypadku, gdy proteza oferuje znacznie więcej funkcji, odpowiednio precyzyjne i skuteczne sterowanie nie jest już takie proste.

Najlepsze rezultaty osiągnięto wykorzystując techniki rozpoznawania obiektów, polegające na tym, że z sygnału mierzonego z mięśni wydobywa się tzw. wektor cech, a na jego podstawie określa się intencje pacjenta, poprzez wskazanie jednej z wcześniej zdefiniowanych klas, do których zalicza się ten wektor. Klasa odpowiada typowi ruchu, jaki proteza powinna wykonać lub stanu, w jakim powinna się znaleźć. Natura wektora cech, jak i sposób przydzielania klasy do wektora cech, mogą różnić się w szerokim zakresie i są przedmiotem badań na całym świecie. Bardziej szczegółowo proces rozpoznawania, wraz z propozycją konkretnej realizacji, omówiony został w rozdziale 4.

Układ sterowania bioprotezą, oprócz informacji o intencjach pacjenta, tj. klasy ruchu, podczas sterowania przegubami może brać pod uwagę także dodatkowe informacje z innych źródeł. Proteza może być wyposażona w różnego rodzaju czujniki, np. poślizgu montowany na końcach palców dający informację o wyslizgiwaniu się chwyconego przedmiotu, czujniki

zbliżeniowe montowane wewnątrz dłoni, czy czujniki siły ścisku. Ponadto, bioprotezę można wyposażyć w akcelerometry i uzależnić zachowanie protezy, np. siłę ścisku, także od szybkości, z jaką porusza się cała proteza.

Dodatkowe czujniki mogą wpływać na zachowanie protezy, które w przypadku zdrowej kończyny można określić, jako zachowania odruchowe (np. mocniejsze chwycenie ciężkiego lub wyslizgującego się przedmiotu). Jednak główna informacja sterująca powinna mieć źródło w świadomych decyzjach pacjenta, co wiąże się z koniecznością jak najskuteczniejszego rozpoznawania intencji pacjenta, tj. klas ruchów lub stanów, na podstawie sygnałów z mięśni.

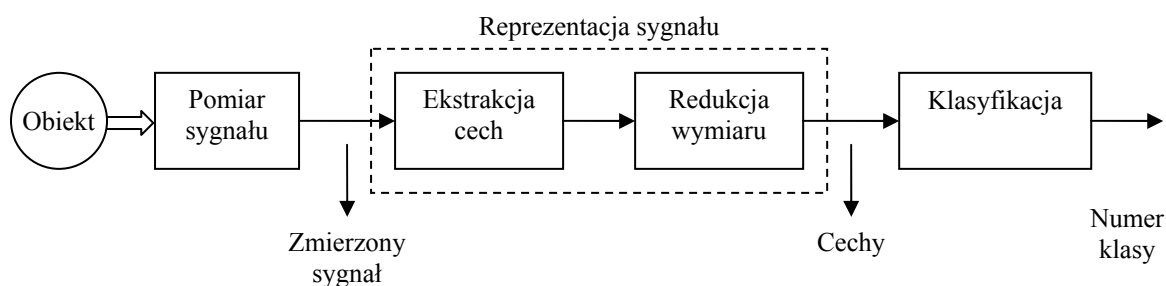
4. Rozpoznawanie obiektów

4.1. Wprowadzenie

Rozpoznawaniem obiektów (z ang. Pattern Recognition) określa się proces, którego zadaniem jest przyporządkowanie obiektu do jednej z wcześniej określonych klas na podstawie zbioru cech tego obiektu i ich wzajemnych relacji. Z rozpoznawaniem spotykamy się na co dzień. Najdoskonalszym systemem rozpoznającym jest ludzki umysł, który dzięki zmysłom odbiera sygnały ze świata otaczającego dotyczące pewnych zjawisk, a następnie łącząc w grupy zjawiska podobne, nadaje im pewne znaczenie (przypisuje klasę). W ten sposób jesteśmy w stanie przeczytać tekst, który był pisany nieznanym dla nas charakterem pisma lub rozpoznać w tłumie znajomą osobę.

W latach pięćdziesiątych, gdy coraz bardziej dostępne były maszyny cyfrowe, zaczęto prowadzić badania nad automatyzacją procesu rozpoznawania. Już na samym początku rozwijania tej dziedziny nauki pojawiły się dwa nurty różniące się podejściem do problemu. Pierwsze podejście, nazywane biocybernetycznym, polega na próbie modelowania systemu nerwowego w ten sposób, aby naśladował procesy umysłowe. Model systemu nerwowego składa się z dużej liczby połączonych ze sobą sztucznych neuronów, które z kolei są bardzo uproszczonymi modelami neuronów rzeczywistych. Drugie podejście polega na poszukiwaniu matematycznych metod rozpoznawania korzystających z modeli statystycznych. Jak okazało się po latach, nurty te nie są rozłączne i często, wychodząc z innych przesłanek, prowadzą do takich samych wyników, np. perceptron Rosenblatta i metoda funkcji potencjałowych (patrz [6]).

W zależności od rodzaju rozpoznawanych obiektów, co wiąże się z rodzajem mierzonych cech i rodzajem klas, do jakich przypisywane są te obiekty, mamy do czynienia z szeroką skalą rodzajów systemów rozpoznawania i ich zastosowań. Należą do nich systemy rozpoznawania mowy, wykrywanie nieprawidłowości w medycznych obrazach i sygnałach czasowych (np. EKG), rozpoznawanie pisma i wiele innych, np. badany w tej pracy system rozpoznawania stanów dłoni na podstawie sygnałów napięciowych mięśni przedramienia. Niezależnie od konkretnego zastosowania, system rozpoznawania obiektów można przedstawić za pomocą kilku bloków funkcyjnych, co zostało przedstawione na Rys. 4.1.1.



Rys. 4.1.1. Schemat blokowy systemu rozpoznawania obiektów

Proces rozpoznawania składa się z czterech etapów. Pierwszym etapem procesu jest pomiar pewnych informacji opisujących obiekt i zamiana ich na postać cyfrową. Od natury tych informacji zależy sposób pomiaru oraz wyposażenie techniczne używane przy pomiarze. W przypadku informacji wizualnych używa się kamer lub skanerów, w przypadku sygnałów dźwiękowych wykorzystuje się mikrofony, a w przypadku sygnałów EMG korzysta się z elektrod oraz wzmacniaczy pomiarowych. Zmierzony sygnał konwertowany jest na postać

cyfrową. Niezależnie od natury informacji ważne jest, aby pomiar odbywał się z największą możliwą wiernością.

Drugi etap polega na wydobyciu ze zmierzonego sygnału informacji, na podstawie których będzie przeprowadzana klasyfikacja obiektu. W praktycznych zadaniach rozpoznawania rzadko zdarza się, aby bezpośrednio zmierzony sygnał miał odpowiednią formę do klasyfikacji. Typowym przykładem są sygnały medyczne np. EMG. W tym przypadku bezpośrednio zmierzony sygnał ma postać ciągu próbek określających przebieg sygnału, ale do klasyfikatora trafiają jedynie pewne cechy tego sygnału, np. widmo częstotliwościowe, co daje lepsze rezultaty rozpoznawania.

W związku z tym, że wektor cech określający dany obiekt może nieść bardzo dużo informacji, często konieczne jest zredukowanie jego rozmiaru. Ważne jest, aby redukcji wymiaru dokonać w sposób, który zachowa w wektorze cech informacje ważne dla procesu klasyfikacji, a usunie informacje nieużyteczne. Powszechnie znane są dwa podejścia do rozwiązania tego problemu:

- selekcja cech – polegająca na wyborze najlepszego możliwego podzbioru oryginalnego zbioru cech,
- projekcja cech – polegająca na określeniu pewnego przekształcenia, które przeprowadza oryginalny wektor cech w wektor o mniejszym wymiarze, w ten sposób, że cechy wynikowe są pewną kombinacją cech oryginalnych.

Otrzymany zredukowany wektor cech trafia do klasyfikatora, którego zadaniem jest przypisanie go do jednej z klas. Istnieje bardzo dużo metod klasyfikacji. Ogólnie można je podzielić na dwie grupy. Do pierwszej grupy należą metody statystyczne, do których zalicza się np. klasyfikator NN (z ang. Nearest Neighbor - najbliższy sąsiad), czy k-NN (k najbliższych sąsiadów). Drugą grupę, która jest ostatnio bardzo popularna, stanowią klasyfikatory neuronowe, z których najbardziej znany to klasyfikator wykorzystujący wielowarstwową sieć neuronową MLP (z ang. Multi Layer Perceptron).

Mimo, że klasyfikator jest najważniejszą częścią systemu rozpoznawania, nie należy zapominać, że ogólna sprawność systemu zależy od wszystkich jego elementów. Nawet najskuteczniejszy klasyfikator nie będzie dawał satysfakcjonujących wyników, jeżeli pomiar sygnałów nie będzie wykonywany poprawnie lub zastosujemy nieodpowiedni sposób reprezentacji wektora cech. Niezastosowanie redukcji rozmiaru wektora cech prowadzi do bardzo powolnego działania systemu, a w przypadku klasyfikatorów neuronowych do znacznego wydłużenia czasu potrzebnego do wytrenowania sieci, co w praktycznych zastosowaniach jest często nie do przyjęcia. Z kolei zbyt duża redukcja wymiaru wektora cech lub przeprowadzenie jej w nieodpowiedni sposób powodują stratę ważnych informacji, co prowadzi do błędów klasyfikacji i obniżenia sprawności całego systemu.

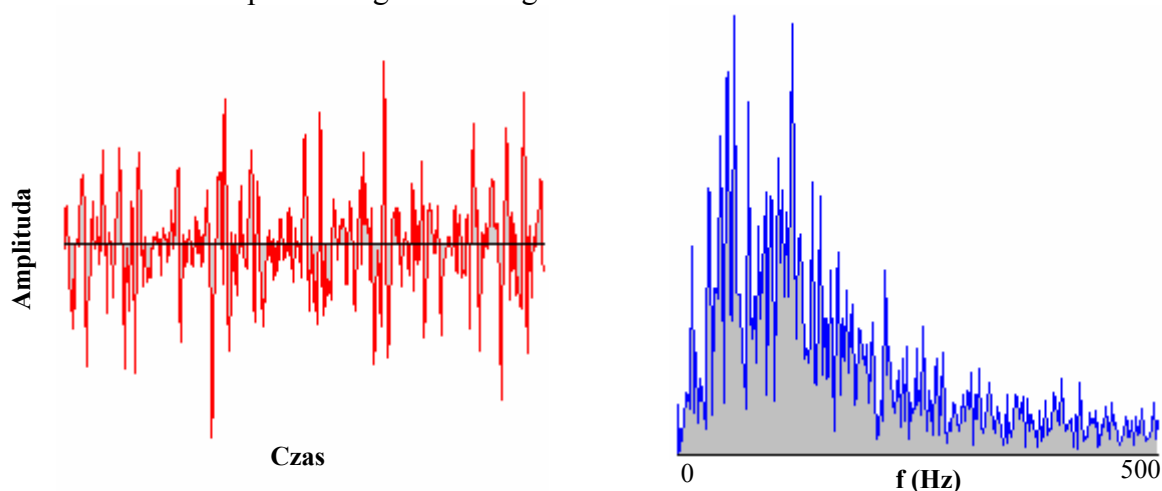
W kolejnych punktach tego rozdziału zaprezentowano realizację poszczególnych etapów procesu rozpoznawania dla badanego problemu rozpoznawania stanów dłoni na podstawie sygnałów miopotencjałów mierzonych na przedramieniu człowieka. Pomiar przeprowadzany jest za pomocą toru pomiarowego składającego się z elektrod EKG oraz zaprojektowanych wzmacniaczy pomiarowych. Po przetworzeniu na postać cyfrową, następuje ekstrakcja cech, którą w tym wypadku jest transformata Fouriera sygnałów. W celu redukcji wymiaru wektora cech stosuje się proste przekształcenie wybierające z widma sygnałów określoną liczbę składowych harmonicznym odpowiednio uśrednionych po sąsiednich harmonicznym. Tak skonstruowany wektor cech trafia do klasyfikatora, który określa na tej podstawie stan dłoni reprezentowany przez numer klasy.

4.2. Pomiar sygnału

4.2.1. Natura sygnałów miopotencjałów

Z niemal każdym organem ciała związany jest sygnał bioelektryczny. Niektóre z nich trudno jest zmierzyć, ale sposoby pomiarów innych są dobrze znane od lat i często stosowane. Najbardziej znanym i powszechnie przeprowadzanym pomiarem sygnału bioelektrycznego jest pomiar aktywności mięśnia serca noszący nazwę elektrokardiogramu (EKG). Pomiar EKG jest bardzo pomocny przy diagnozie zaburzeń rytmu, uszkodzeń po zawale i wielu innych chorób serca. Ogólnie sygnał bioelektryczny dowolnego mięśnia nosi nazwę elektromiogramu (EMG). Sygnał EMG może być mierzony na dwa sposoby. Pierwszy sposób to pomiar inwazyjny za pomocą elektrod igłowych. Realizacja takich pomiarów możliwa jest jedynie w warunkach laboratoryjnych. Wynikiem pomiaru jest sygnał pochodzący z pojedynczego mięśnia. Drugi sposób pomiaru to pomiar za pomocą elektrod powierzchniowych. Pomiar taki jest znacznie łatwiejszy w realizacji i znalazł zastosowanie nie tylko w medycynie.

Sygnał zmierzony powierzchniowo pochodzi z całej grupy mięśni, przy czym zsumowane sygnały z poszczególnych mięśni podlegają filtracji przez okalającą tkankę tłuszczową oraz skórę. Stąd amplituda sygnału wynikowego ma naturę stochastyczną i silnie zależy od aktywności mięśniowej oraz od miejsca umocowania elektrody. Wartość międzyszczytowa amplitudy (peak-to-peak) może wynosić do 10 mV, wartość skuteczna (RMS) do 1,5 mV. Sygnał użyteczny (o energii przekraczającej energię szumu elektrycznego) zawarty jest w przedziale od 0 do 500 Hz, przy czym dominująca część energii występuje w przedziale od 50 do 150 Hz. Na Rys. 4.2.1 przedstawiono przykład sygnału EMG oraz jego widmo zmierzone podczas zginania nadgarstka.



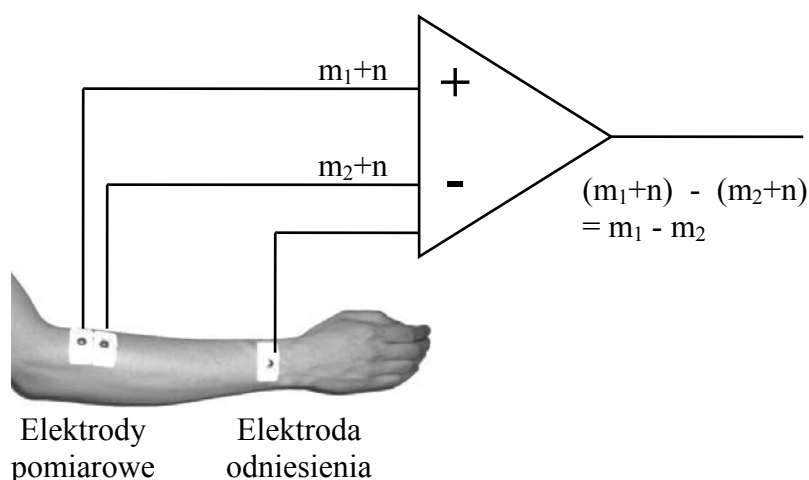
Rys. 4.2.1. Sygnał EMG oraz jego widmo częstotliwościowe zmierzone na przedramieniu podczas zginania nadgarstka

4.2.2. Pomiar miopotencjałów

Przy pomiarze sygnałów elektrycznych mięśni bardzo ważnym czynnikiem jest zapewnienie dobrego połączenia układu pomiarowego ze skórą. W tym celu stosuje się odpowiednio skonstruowane elektrody. Istnieje wiele rodzajów elektrod różniących się budową i materiałem, z którego są wykonane. Ze względu na dobre właściwości (patrz [10]), do budowy elektrod służących do pomiaru sygnałów z mięśni najczęściej stosuje się srebro w połączeniu z chlorkiem srebra (Ag/AgCl). Aby polepszyć kontakt skóra-elektroda można dodatkowo zastosować żel przewodzący. Na rynku dostępny jest szeroki wybór jednorazowych elektrod EKG, które można stosować także do pomiarów z innych mięśni.

Podczas pomiarów EMG należy zwrócić uwagę na kilka ważnych zjawisk, z których najważniejszym są szumy. Dwoma najsilniejszymi źródłami szumów są aparatura pomiarowa oraz otoczenie. W przypadku aparatury pomiarowej szum pochodzi ze wszystkich elementów i nie można go wyeliminować, jest to tzw. szum własny układu pomiarowego. Możliwa jest jedynie jego redukcja poprzez zastosowanie elementów elektronicznych wysokiej jakości i dobre zaprojektowanie układu pomiarowego. Drugi rodzaj szumów powodowany jest promieniowaniem elektromagnetycznym występującym w otoczeniu. Źródłem tego promieniowania są praktycznie wszystkie urządzenia elektryczne, przewody energetyczne i np. nadajniki telewizyjne. Szum elektryczny otoczenia występuje wszędzie i może mieć amplitudę przewyższającą amplitudę sygnałów EMG nawet o kilka rzędów. Największe zakłócenia powodowane są przez źródła zasilania i mają częstotliwość ok. 50 Hz. Inne rodzaje szumów (patrz [8]) mają już znacznie mniejszy wpływ na pomiar.

W celu wyeliminowania wyżej opisanego zjawiska stosuje się pomiar różnicowy. Polega on na tym, że mierzy się sygnały w dwóch punktach, a następnie odejmuje się je od siebie i wzmacnia. W ten sposób część wspólna sygnałów, w której skład wchodzi szumy elektryczne otoczenia, jest usuwana, a wzmacniany jest jedynie sygnał stanowiący różnicę między sygnałami z dwóch punktów pomiarowych, zawierający sygnał pochodzący z mięśni. Ideę pomiaru różnicowego przedstawiono na Rys. 4.2.2.



Rys. 4.2.2. Schemat pomiaru różnicowego aktywności mięśni

W praktyce wykonanie operacji różnicy sygnałów w sposób idealny jest niemożliwe. Dokładność, z jaką wzmacniacz różnicowy odejmuje sygnały, jest określana przez współczynnik tłumienia sygnału sumacyjnego CMRR (z ang. Common Mode Rejection Ratio) wyrażany w decybelach. CMRR definiuje się jako stosunek wzmocnienia sygnału różnicowego do wzmocnienia sygnału wspólnego. Ogólnie CMRR powinien być możliwie jak największy, przy czym wystarczająca wartość to ok. 90 dB.

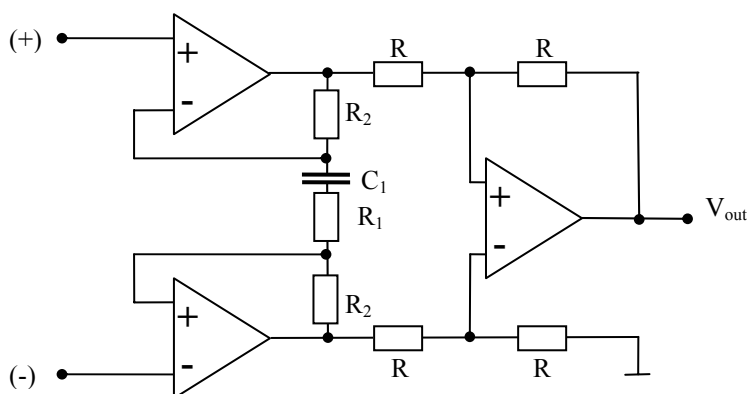
Kolejnym ważnym czynnikiem mającym wpływ na pomiar EMG jest impedancja wejściowa wzmacniacza. W zależności m.in. od stanu skóry (wilgotności), impedancja połączenia skóra-elektroda może wahać się w granicach od kilku kiloomów do kilku megaomów. Ważne jest, aby impedancja wejściowa wzmacniacza różnicowego była jak największa, aby nie obciążać wejścia, co może prowadzić do zniekształceń mierzonego sygnału. Dzisiejsze wzmacniacze pomiarowe mają impedancję rzędu $10^{12} \Omega$ przy 5 pF łączonych równolegle, co jest wartością wystarczającą.

Z powodu tego, że główne szumy układu pomiarowego indukują się w przewodach łączących wejścia wzmacniacza z elektrodami, a ponadto na wejściu pojawia się takie zjawisko jak sprzężenie pojemnościowe (patrz [8]), w wielu pracach sugeruje się, aby układ pomiarowy był zaprojektowany w ten sposób, że elektrody położone są możliwie blisko

wzmacniacza pomiarowego. Rozwiązanie takie znane jest jako aktywna elektroda. Sygnał wyjściowy z takiego układu jest już wzmocniony i zakłócenia pochodzące z otoczenia nie mają na niego znaczącego wpływu.

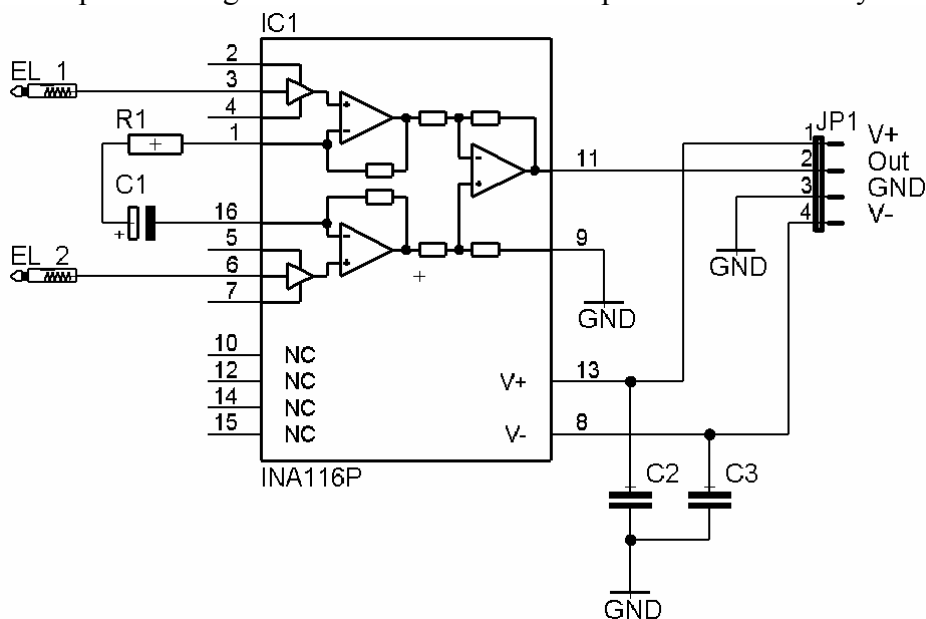
4.2.3. Projekt wzmacniacza pomiarowego

Do celów badawczych zaprojektowano układ pomiarowy składający się z elektrod EKG oraz wzmacniaczy pomiarowych spełniających wymagania przedstawione w punkcie 4.2.2. Typowy układ pomiarowy stosowany do wzmacniania sygnałów biomedycznych (zaczepnięty z [11]) przedstawiony został na Rys. 4.2.3. Charakteryzuje się on bardzo dużą impedancją wejściową oraz dużym współczynnikiem CMRR. Wielkość wzmocnienia układu ustala się dobierając wartość rezystora R_1 .



Rys. 4.2.3. Schemat typowego wzmacniacza stosowanego przy pomiarach sygnałów biomedycznych

Dzięki swoim parametrom konfiguracja tego typu jest często stosowana i doczekała się licznych realizacji w postaci układów scalonych. Przykładem takiego układu jest zastosowany w pracy układ INA116 firmy Burr-Brown, którego impedancja wejściowa wynosi $10^{15} \Omega / 0.2 \text{ pF}$, a współczynnik CMRR 94 dB przy wzmocnieniu ustalonym na 1000. Schemat układu pomiarowego z zastosowaniem INA116 przedstawiono na Rys. 4.1.1.

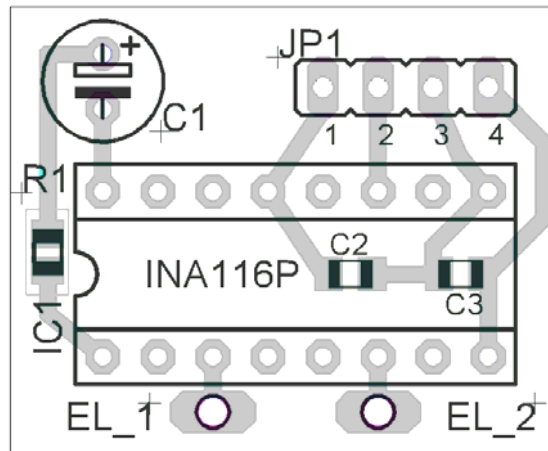


Rys. 4.2.4. Schemat układu pomiarowego

Oprócz rezystora R_1 ustalającego wzmocnienie oraz kondensatora C_1 połączonego szeregowo z rezystorem, zgodnie ze specyfikacją [12] układ wymaga jedynie podłączenia kondensatorów odprzegających między zasilaniem i masą o wartościach $0,1 \mu\text{F}$. Wartość rezystancji R_1 zależy od wartości wzmocnienia, jakiego oczekujemy. Dla wzmocnienia 1000 R_1 powinno wynosić 50Ω . Przy ustalonym R_1 od wartości C_1 zależy dolna częstotliwość graniczna pasma przenoszenia wzmacniacza. Wyznaczyć ją można korzystając ze wzoru

$$f_d = \frac{1}{2\pi R_1 C_1}. \quad (4.1)$$

Jak łatwo zauważyć, im większa jest pojemność C_1 , tym niższa częstotliwość graniczna. Dla $R_1 = 50 \Omega$ (wzmocnienie 1000) oraz $C_1 = 1000 \mu\text{F}$ częstotliwość f_d wynosi 3,2 Hz, co jest wartością w pełni wystarczającą.

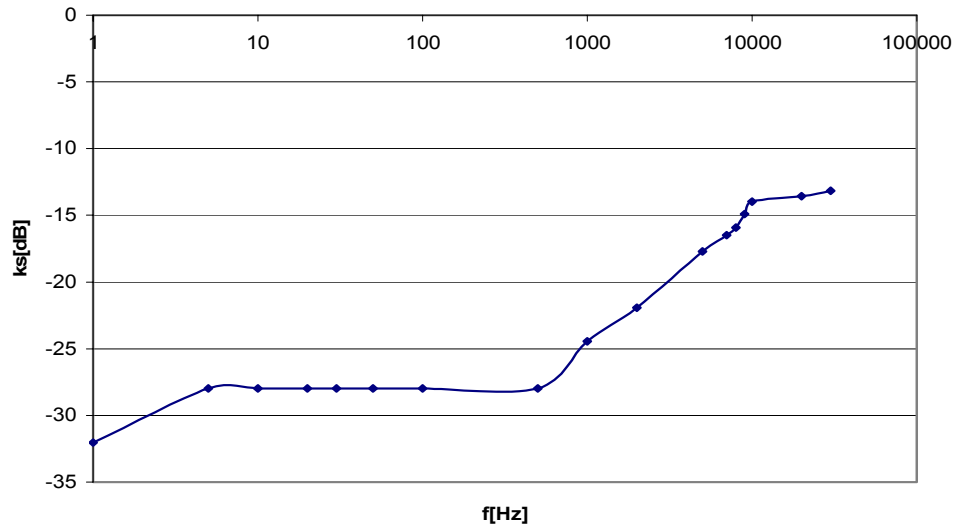


Rys. 4.2.5. Schemat płytki drukowanej układu pomiarowego

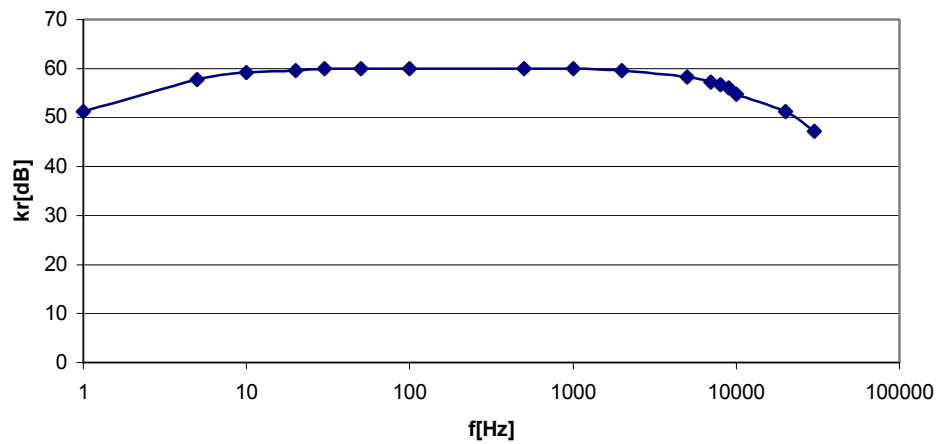
Projekt płytki drukowanej, wraz z elementami przedstawiony został na Rys. 4.2.5. Jako R_1 , C_2 oraz C_3 zastosowano elementy w wersji montowanej powierzchniowo, co pozwoliło zmniejszyć rozmiar płytki. Ze względu na wymaganą dużą pojemność, jako kondensator C_1 zastosowano element montowany tradycyjnie.

4.2.4. Parametry układu pomiarowego

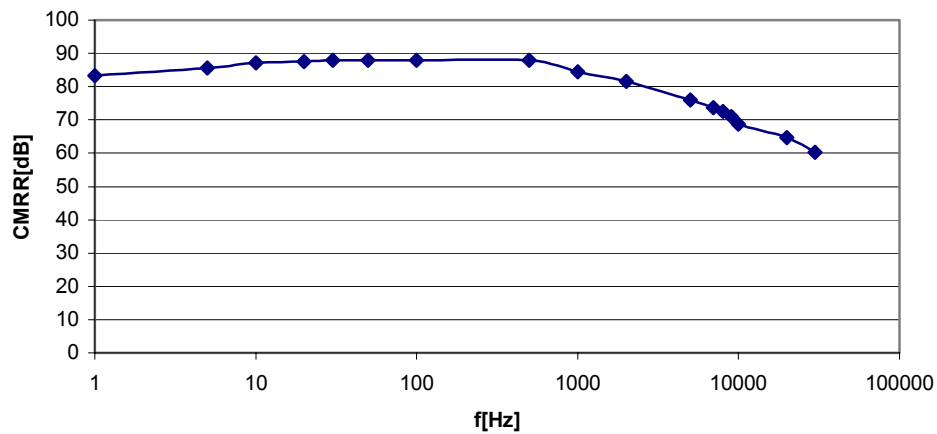
Poniżej przedstawiono zmierzone charakterystyki wykonanego układu pomiarowego obejmujące wykres wzmocnienia sumacyjnego, różnicowego oraz współczynnika CMRR w funkcji częstotliwości.



Rys. 4.2.6. Wzmocnienie sygnału sumacyjnego k_s w funkcji częstotliwości



Rys. 4.2.7. Wzmocnienie sygnału różnicowego k_r w funkcji częstotliwości



Rys. 4.2.8. Współczynnik tłumienia sygnału sumacyjnego CMRR w funkcji częstotliwości

Jak widać na powyższych wykresach, w najbardziej interesującym przedziale częstotliwości (do 500 Hz) układ wzmacniacza charakteryzuje się współczynnikiem CMRR o wartości od 83 dB dla 1 Hz do 88 dB dla częstotliwości 500 Hz. Uwzględniając bardzo dużą impedancję wejściową oraz niskie szумы własne, jakimi charakteryzuje się układ INA116, można stwierdzić, że zaprojektowany układ spełnia wszystkie stawiane wymagania.

4.2.5. Konwersja na postać cyfrową

Kolejnym etapem przetwarzania sygnałów EMG jest zamiana zmierzonego sygnału analogowego na postać cyfrową. Proces ten dokonywany jest przy użyciu tzw. przetwornika analogowo-cyfrowego (A/C). Konwersja na postać cyfrową polega na zamianie ciągłego sygnału $g(t)$ na dyskretną serię próbek wyrażonych w postaci szeregu liczb. Przykład sygnału ciągłego oraz jego postać dyskretna przedstawione są na Rys. 4.2.9.

Amplituda kolejnych próbek sygnału g mierzona jest co określony odstęp czasu Δt , nazywany okresem próbkowania. Odwrotność okresu próbkowania nazywana jest częstotliwością próbkowania i wyrażona jest w hercach. Zmierzony ciąg próbek stanowi reprezentację cyfrową sygnału analogowego. Przedstawić to można następująco

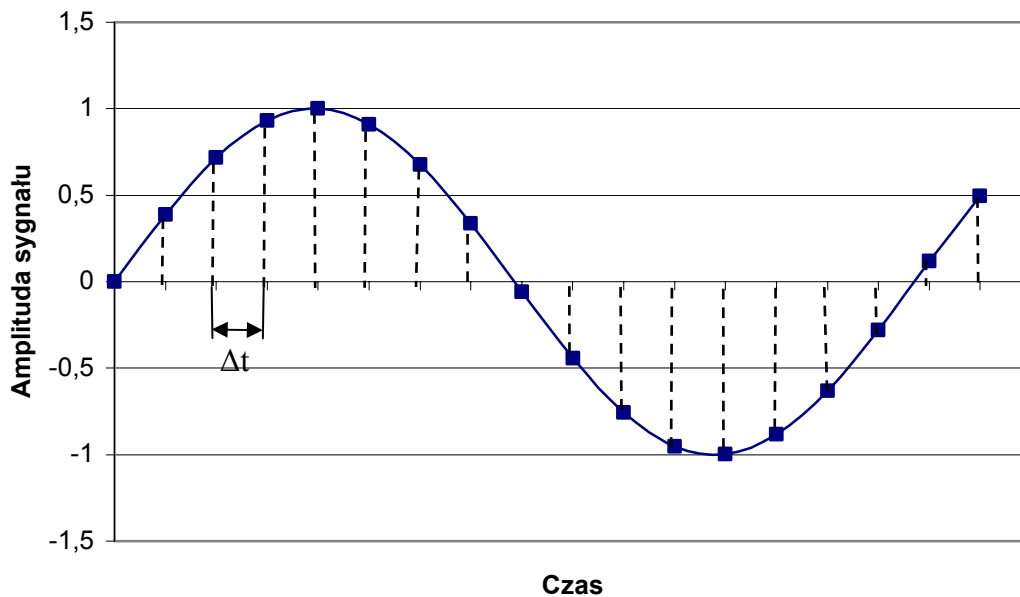
$$g(t) \rightarrow \{g(0), g(\Delta t), g(2\Delta t), \dots, g(k\Delta t), \dots\} = \{g[0], g[1], g[2], \dots, g[k], \dots\}, \quad (4.2)$$

gdzie $g(t)$ – sygnał analogowy, $g[k]$ – k -ta próbka sygnału $g(t)$.

Dokładność pomiaru amplitudy określona jest przez rozdzielczość przetwornika. Rozdzielczość zależy od ilości bitów reprezentujących wartość amplitudy próbki. Ogólnie przetwornik n -bitowy jest w stanie rozróżnić 2^n poziomów, co dla przetwornika 16 bitowego daje 65536 różnych poziomów.

Oprócz częstotliwości próbkowania (regulowanej) i rozdzielczości, do podstawowych parametrów przetwornika zalicza się zakres napięć wejściowych. W przypadku 16 bitowego przetwornika o zakresie napięć wejściowych $(-1,1)$ V rozdzielczość napięciowa dla 1 bitu wynosi

$$2V / 2^{16} = 30,5 \mu V.$$



Rys. 4.2.9. Przykład sygnału analogowego i jego postaci próbkowanej

Bardzo ważnym zagadnieniem jest dobór częstotliwości próbkowania sygnału. Częstotliwość powinna być wybrana w ten sposób, aby umożliwić poprawną reprodukcję informacji analogowej. Według kryterium Shannona (patrz [9]), częstotliwość próbkowania f powinna być co najmniej dwa razy większa od największej częstotliwości składowej sygnału. Oznacza to, że przy próbkowaniu z częstotliwością f maksymalna częstotliwość składowa sygnału, która będzie poprawnie reprezentowana wynosi $f/2$. W przypadku pomiarów EMG, biorąc pod uwagę, że sygnał użyteczny zawarty jest w paśmie 0-500 Hz, częstotliwość próbkowania nie powinna być mniejsza niż 1 kHz.

4.3. Ekstrakcja cech

Jak zaznaczono w punkcie 4.1, rzadko zdarza się, aby informacja bezpośrednio zmierzona trafiała do klasyfikatora, w praktyce często należy wydobyć z tej informacji pewne charakterystyczne cechy, na podstawie których obiekt będzie klasyfikowany. W przypadku sygnałów czasowych, do których zalicza się EMG, jednym z podstawowych i często stosowanych zestawów cech jest widmo częstotliwościowe sygnału. Widmo niesie informacje o zawartości poszczególnych składowych częstotliwościowych w sygnale, co zależy m.in. od aktywności poszczególnych mięśni, których dotyczy pomiar, a więc od stanu dłoni. Widmo częstotliwościowe sygnału otrzymuje się przez zastosowanie transformaty Fouriera.

4.3.1. Transformata Fouriera

Podstawą analizy częstotliwościowej jest twierdzenie Fouriera (patrz [9]) mówiące o tym, że dowolny sygnał o nieskończenie długim czasie trwania można rozłożyć na szereg nieskończenie długich w czasie składowych sinusoidalnych o odpowiednich amplitudach, częstotliwościach i fazach. Prawdziwe jest również twierdzenie odwrotne, tzn. dowolny sygnał o nieskończenie długim czasie trwania można zsyntezować ze składowych sinusoidalnych o określonych amplitudach, częstotliwościach i fazach. Parametry te, dla dowolnego ciągłego sygnału, pozwala wyznaczyć transformata Fouriera, przy czym dotyczy ona sygnałów o nieskończonej długości i opiera się na całkowaniu sygnału analogowego. W przypadku sygnałów dyskretnych używa się tzw. dyskretnej transformaty Fouriera, która zamiast operacji całkowania stosuje operacje sumy. Dyskretna transformata Fouriera opisana jest wzorem

$$G[k] = \sum_{n=0}^{N-1} g[n] \exp\left(-i \frac{2\pi}{N} kn\right), \quad (4.3)$$

gdzie $g[n]$ jest n -tą próbką zmierzonego sygnału, N – całkowitą liczbą próbek, a $G(k)$ – k -tą składową transformaty, przy czym $k = 0, \dots, N-1$.

Na podstawie transformaty Fouriera można wyznaczyć widmo częstotliwościowe sygnału, poprzez obliczenie modułu dla każdej obliczonej składowej zespolonej. Ponadto wybiera się jedynie połowę wyznaczonych składowych, tzn. przyjmuje się $k = 0, \dots, N/2$, co wynika z faktu, że transformata Fouriera jest symetryczna względem indeksu $N/2$ i pozostałe składowe mają taką samą amplitudę.

Jeśli przez $\Delta f = f/N$ opiszemy rozdzielczość częstotliwościową, gdzie f jest częstotliwością próbkowania, a N liczbą próbek, to pierwsza składowa widma $G(0)$ zawiera składową stałą sygnału mierzonego, $G(1)$ zawiera składową o częstotliwości Δf , a każda następna jej wielokrotność, tzn. $G(k)$ zawiera składową o częstotliwości $k\Delta f$. Przykład sygnału spróbkowanego z częstotliwością 1000 Hz oraz jego widmo zostały przedstawione wcześniej na Rys. 4.2.1.

Jak zauważono wcześniej, transformata Fouriera dotyczy sygnałów o nieskończonej długości, jednak w przypadku sygnałów próbkowanych dostępna jest tylko pewna skończona liczba próbek. Transformata Fouriera zastosowana do wybranego fragmentu sygnału traktuje ten fragment jako jeden pełen okres sygnału. Niesie to za sobą pewne konsekwencje. Jeżeli analizowany fragment zawiera niecałkowitą liczbę cykli sygnału periodycznego, to po połączeniu fragmentów powstaną nieciągłości, co wpływa na widmo sygnału. Rozwiązaniem tego problemu jest zastosowanie tzw. okienkowania sygnału. Okienkowanie polega na zmniejszaniu stopnia nieciągłości sygnału na jego krańcach poprzez przemnożenie próbek przez pewną funkcję dążącą na krańcach do zera. Funkcja okna W może mieć różną postać, przy czym w przypadku, gdy nie stosuje się okienkowania sygnału, mówi się o tzw. oknie prostokątnym. Najbardziej znane i najczęściej stosowane okna to:

- okno Gaussa

$$W[n] = \exp\left(-2\sigma^2\left(\frac{n}{N} - \frac{1}{2}\right)^2\right), \quad (4.4)$$

gdzie σ jest parametrem, typowa wartość to $\sigma = 4,24$,

- okno Hanninga

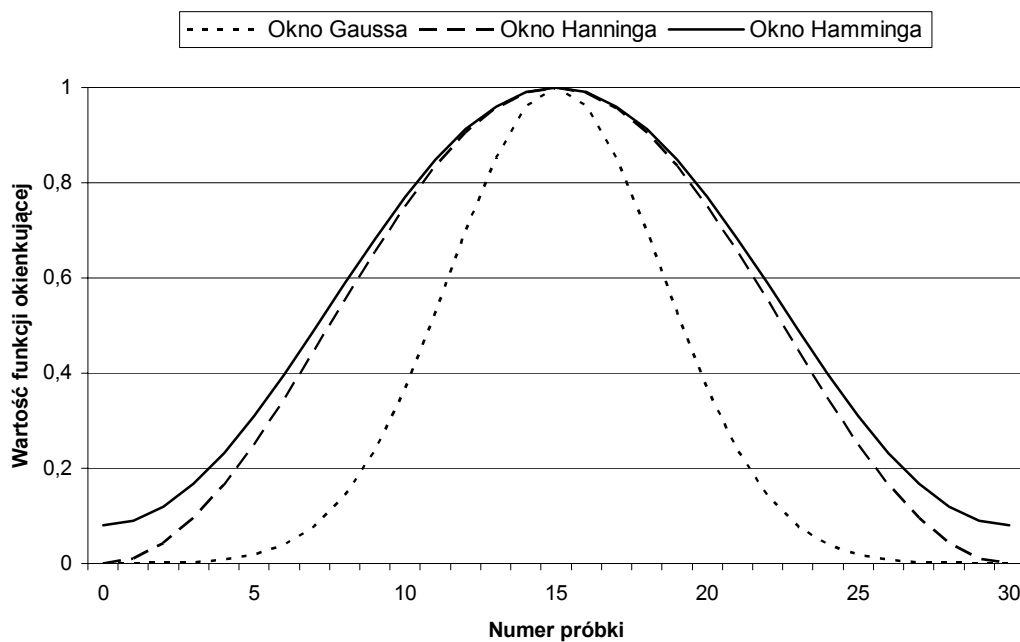
$$W[n] = 0,5 - 0,5 \cos\left(2\pi \frac{n}{N}\right), \quad (4.5)$$

- okno Hamminga

$$W[n] = 0,54 - 0,46 \cos\left(2\pi \frac{n}{N}\right). \quad (4.6)$$

W powyższych wzorach przyjęto: W – funkcja okienkująca, N – liczba próbek w oknie, n – numer próbki w oknie, $n = 0, \dots, N-1$.

Postać okien dla fragmentu sygnału o 30 próbkach przedstawiona jest na Rys. 4.3.1.



Rys. 4.3.1. Postać funkcji okienkujących dla okna sygnału o 30 próbkach

Dyskretna transformata Fouriera wykorzystująca inne okno niż okno prostokątne opisana jest wzorem

$$G[k] = \sum_{n=0}^{N-1} W[n]g[n]\exp\left(-i\frac{2\pi}{N}kn\right), \quad (4.7)$$

gdzie W jest określone przez wybrany rodzaj funkcji okna.

Wyznaczone w ten sposób składowe harmoniczne stanowią pewien zbiór cech charakteryzujących sygnał. Należy pamiętać, że w przypadku próbkowania w oknach o wielkości np. 256 próbek otrzymuje się 128 harmonicznych (niezależnie od częstotliwości próbkowania), co może być zbyt dużą ilością danych dla klasyfikatora, dlatego konieczna jest redukcja wymiaru wektora cech.

4.3.2. Inne sposoby ekstrakcji cech

Transformata Fouriera jest podstawowym i często stosowanym narzędziem analizy sygnałów, lecz nie w każdym przypadku jest to odpowiednia metoda, dlatego że określa jedynie, jakie składowe harmoniczne pojawiły się w danym oknie sygnału, a nie mówi nic o lokalizacji tych składowych w czasie. Na podstawie transformaty Fouriera nie jesteśmy w stanie określić, czy poszczególne składowe występowały w sygnale jednocześnie, czy kolejno po sobie. Aby polepszyć lokalizację czasową składowych, można zmniejszyć rozmiar okna wycinającego sygnał, ale powoduje to automatycznie zmniejszenie rozdzielczości częstotliwościowej. Przykładowo, dla sygnału próbkowanego z częstotliwością 1000 Hz, jeśli wytniemy okno tego sygnału o 512 próbkach (około 0,5 sekundy), to możemy wyznaczyć z niego 256 składowych częstotliwościowych, ale jeśli zwiększymy czterokrotnie rozdzielczość czasową stosując okno o 128 próbkach, to rozdzielczość częstotliwościowa spada i możemy wyznaczyć tylko 64 składowe. Ogólnie, im większa rozdzielczość czasowa, tym mniejsza rozdzielczość częstotliwościowa. Z tego powodu transformata Fouriera jest nienajlepszą metodą analizy sygnałów, których struktura częstotliwościowa szybko zmienia się w czasie.

Rozwiązaniem wyżej opisanego problemu jest zastosowanie jednej z metod przetwarzania sygnału w połączonej dziedzinie czasu i częstotliwości. Jedną z najnowszych i intensywnie badanych metod jest tzw. analiza falkowa (z ang. wavelets). Transformata Fouriera rozwija sygnał na funkcje elementarne będące kolejnymi harmonicznymi funkcji sinusoidalnych. W przypadku analizy falkowej, sygnał rozwijany jest na zestaw elementarnych funkcji skalowanych w czasie nazywanych falkami. Analiza falkowa wykazuje się dobrą rozdzielczością częstotliwościową dla niskich częstotliwości oraz dobrą rozdzielczością czasową dla częstotliwości wysokich, jednak jej teoria jest bardzo obszerna i zawiła matematycznie, dlatego metoda ta nie została wykorzystana w niniejszej pracy. Niewątpliwie jest to jednak kierunek, w jakim należałoby prowadzić dalsze badania nad wykorzystaniem EMG do sterowania.

4.4. Redukcja wymiaru wektora cech

Jak zaznaczono w rozdziale 4.3.1, rozmiar wektora cech otrzymanego przez zastosowanie transformaty Fouriera może być zbyt duży dla potrzeb klasyfikacji. Ponadto widmo jest bardzo wrażliwe na małe zmiany sygnału mierzonego. Nawet dla ustalonych stanów dłoni pojawiają się w nim fluktuacje utrudniające proces nauki. W celu rozwiązania tych problemów zastosowano trójetapowy proces redukcji rozmiaru wektora cech. W etapie pierwszym widmo jest stabilizowane w czasie, w etapie drugim i trzecim następuje wydobywanie z widma cech będących średnimi harmonicznymi w pewnych zakresach, zgodnie z metodą zaproponowaną w pracy [2].

Etap pierwszy zrealizowano używając reguły uaktualniającej uśrednione widmo \mathbf{G}' w zależności od różnicy między uśrednionym widmem \mathbf{G}' w poprzedzającej chwili, a aktualnym widmem rzeczywistym \mathbf{G} . Reguła ta przyjmuje postać:

$$\mathbf{G}'(n) = \mathbf{G}'(n-1) + \gamma(\mathbf{G}(n) - \mathbf{G}'(n-1)), \quad (4.8)$$

gdzie parametr $\gamma = (0,1]$ jest współczynnikiem korekcji. Dla $\gamma = 1$ uśrednianie nie występuje, tzn. zachodzi $\mathbf{G}'(n) = \mathbf{G}(n)$. Im mniejsza wartość γ , tym mniejszy wpływ mają chwilowe zmiany wartości w widmie rzeczywistym, a co za tym idzie, tym bardziej stałe jest widmo uśrednione. Zastosowanie operacji uśredniania w czasie wprowadza pewne opóźnienie w pracy systemu rozpoznawania, ale powoduje także, że cały system jest bardziej odporny na fluktuacje widma.

Drugi etap redukcji rozmiaru wektora cech polega na uśrednieniu harmonicznym widma względem sąsiednich harmonicznym. Etap ten jest ściśle związany z etapem trzecim, w którym następuje wybranie pewnych harmonicznym z widma wygładzonego. Przy założeniu, że widmo zawiera N harmonicznym, wygładzanie przebiega zgodnie z zależnością

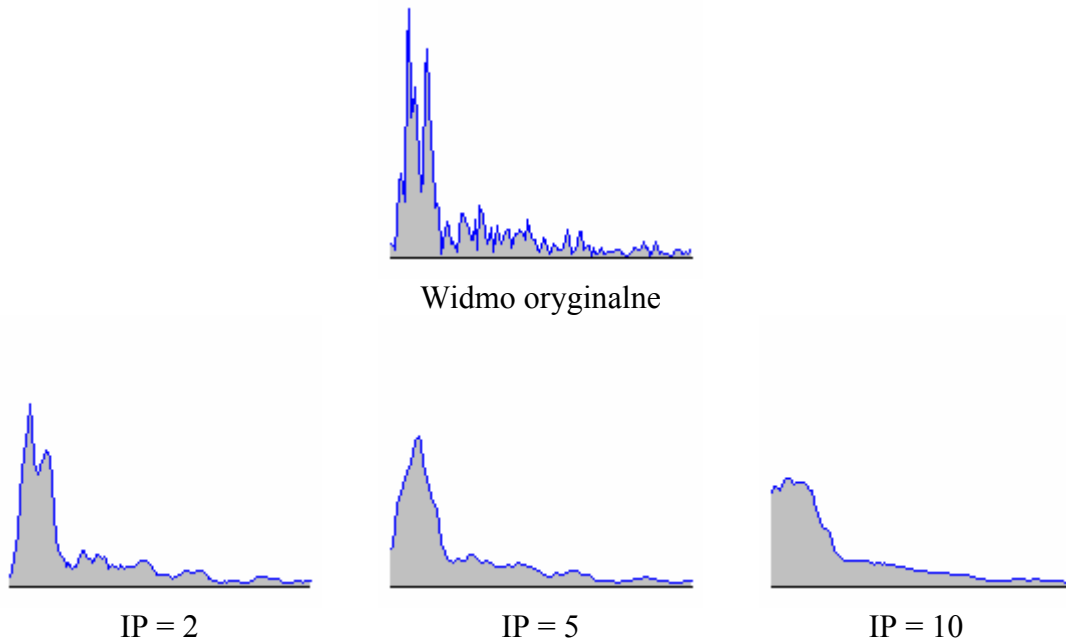
$$S[k] = \sum_{i=IPS}^{IPE} \frac{G'[i]}{IPE - IPS + 1}, \quad k = 0, 1, \dots, N-1,$$

gdzie

$$IPS = \begin{cases} 0 & \text{jeżeli } k - IP < 0 \\ k - IP & \text{w przeciwnym przypadku} \end{cases}, \quad (4.9)$$

$$IPE = \begin{cases} N-1 & \text{jeżeli } k + IP > N-1 \\ k + IP & \text{w przeciwnym przypadku} \end{cases}.$$

W powyższych wzorach IP jest liczbą punktów wygładzających określającą stopień wygładzenia widma. Przykład wygładzania widma dla różnych wartości parametru IP przedstawiono na rysunku Rys. 4.4.1.



Rys. 4.4.1. Przykład widma o 128 harmonicznym i jego uśrednienie dla IP równego 2, 5 oraz 10

Trzecim etapem procesu zmniejszania rozmiaru wektora cech jest selekcja odpowiednich uśrednionych harmonicznych z wygładzonego widma. Jednym z najprostszych sposobów wybierania jest selekcja liniowa dokonywana zgodnie ze wzorem

$$\mathbf{y}[i] = \mathbf{S} \left[i \frac{N-1}{E-1} \right], \quad (4.10)$$

gdzie $i = 0, 1, \dots, E-1$, \mathbf{S} jest wektorem wygładzonego widma o rozmiarze N , a \mathbf{y} wektorem wybranych cech o rozmiarze E , przy czym zakłada się, że oba wektory indeksowane są od zera.

Wektor \mathbf{y} jest wektorem cech pochodzących z jednego kanału pomiarowego. Przyjmując oznaczenie \mathbf{y}^i dla wektora cech z kanału i -tego oraz łącząc cechy ze wszystkich kanałów, otrzymuje się wektor \mathbf{x} o rozmiarze d i postaci

$$\mathbf{x} = (\mathbf{y}^1[0], \dots, \mathbf{y}^1[E-1], \mathbf{y}^2[0], \dots, \mathbf{y}^2[E-1], \dots, \mathbf{y}^L[0], \dots, \mathbf{y}^L[E-1]), \quad (4.11)$$

gdzie L jest liczbą kanałów, E liczbą cech wybranych z jednego kanału oraz $d = LE$.

4.5. Klasyfikacja

4.5.1. Pojęcia podstawowe

Jak wspomniano w punkcie 4.1, zadaniem klasyfikatora jest przypisanie numeru klasy do zadanego wektora cech. Przyjmuje się, że wektor cech \mathbf{x} jest wektorem liczb rzeczywistych o rozmiarze d , tzn. $\mathbf{x} = (x_1, \dots, x_d)$ oraz $\mathbf{x} \in X \subseteq R^d$, gdzie X jest przestrzenią cech. Numer klasy j , do której należy \mathbf{x} , przyjmuje jedną z wartości ze zbioru $M = \{1, 2, \dots, c\}$.

Algorytm klasyfikacji można zdefiniować jako odwzorowanie

$$\psi(\mathbf{x}) = i, \quad (4.12)$$

gdzie $i \in M$ jest wynikiem klasyfikacji. Inaczej mówiąc, algorytm ψ odwzorowuje przestrzeń cech X w zbiór numerów klas M

$$\psi : X \rightarrow M. \quad (4.13)$$

Równoważnym opisem algorytmu klasyfikacji jest rozkład przestrzeni cech na tzw. obszary decyzyjne. Obszary decyzyjne, to obszary przestrzeni cech, oddzielone od siebie powierzchniami decyzyjnymi, dla których zachodzi następująca zależność

$$D_x^{(i)} = \{\mathbf{x} \in X : \psi(\mathbf{x}) = i\} \text{ dla } i \in M. \quad (4.14)$$

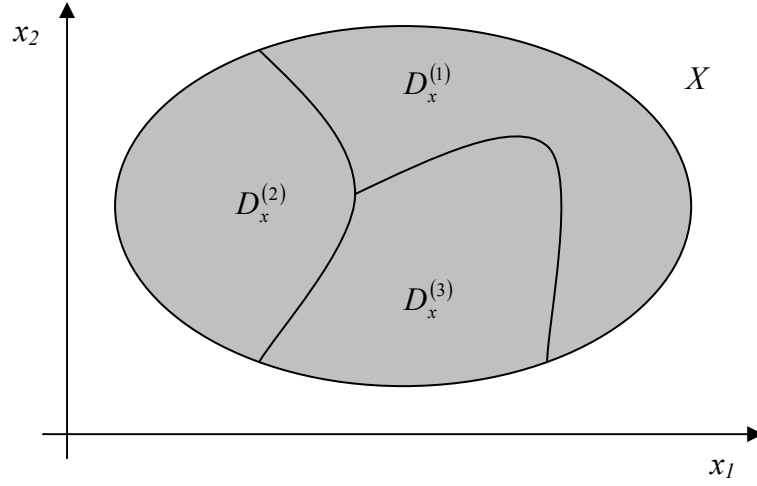
Obszary decyzyjne dzielą przestrzeń cech na rozłączne zbiory pokrywające całą przestrzeń, dlatego zachodzi

$$D_x^{(i)} \cap D_x^{(j)} = \emptyset \text{ dla każdego } i, j \in M, \text{ przy czym } i \neq j$$

oraz

$$\bigcup_{i \in M} D_x^{(i)} = X. \quad (4.15)$$

Należy zauważyć, że obszary decyzyjne jednoznacznie definiują algorytm klasyfikacji. Przykład podziału dwuwymiarowej przestrzeni cech na obszary decyzyjne dla trzech klas przedstawiony został na Rys. 4.5.1.



Rys. 4.5.1. Przykład podziału przestrzeni cech X na obszary decyzyjne

4.5.2. Klasyfikator Bayesowski

Stochastycznie, para (\mathbf{x}, j) jest realizacją pary zmiennych losowych (\mathbf{X}, J) . Prawdopodobieństwo *a priori* klasy j oznacza się przez

$$P(J = j) = p_j \text{ dla } j \in M. \quad (4.16)$$

Ponadto zachodzi

$$\sum_{j=1}^c p_j = 1. \quad (4.17)$$

Warunkowa gęstość cech w klasie, czyli funkcja gęstości prawdopodobieństwa zmiennej losowej \mathbf{X} dla każdej z klas, przyjmuje postać

$$f(\mathbf{x} / j) = f_j(\mathbf{x}) \text{ dla } \mathbf{x} \in X. \quad (4.18)$$

Bezwarunkowa gęstość prawdopodobieństwa zmiennej \mathbf{X} opisana jest wzorem

$$f(\mathbf{x}) = \sum_{j=1}^c p_j f_j(\mathbf{x}). \quad (4.19)$$

Dodatkowo, prawdopodobieństwo warunkowe, że dany obiekt należy do klasy j , nazywane prawdopodobieństwem *a posteriori* klasy j -tej, przyjmuje postać

$$p_j(\mathbf{x}) = P(\mathbf{J} = j / \mathbf{X} = \mathbf{x}), \text{ gdzie } j \in M, \mathbf{x} \in X. \quad (4.20)$$

Prawdopodobieństwo to można wyliczyć na podstawie wzoru Bayesa otrzymując

$$p_j(\mathbf{x}) = \frac{p_j f_j(\mathbf{x})}{f(\mathbf{x})}. \quad (4.21)$$

W przypadku pełnej znajomości wszystkich powyższych prawdopodobieństw i gęstości warunkowych optymalny algorytm Bayesa przyjmuje postać

$$\psi^*(\mathbf{x}) = i, \text{ gdy } p_i(\mathbf{x}) = \max_{k \in M} p_k(\mathbf{x}) \quad (4.22)$$

lub równoważnie

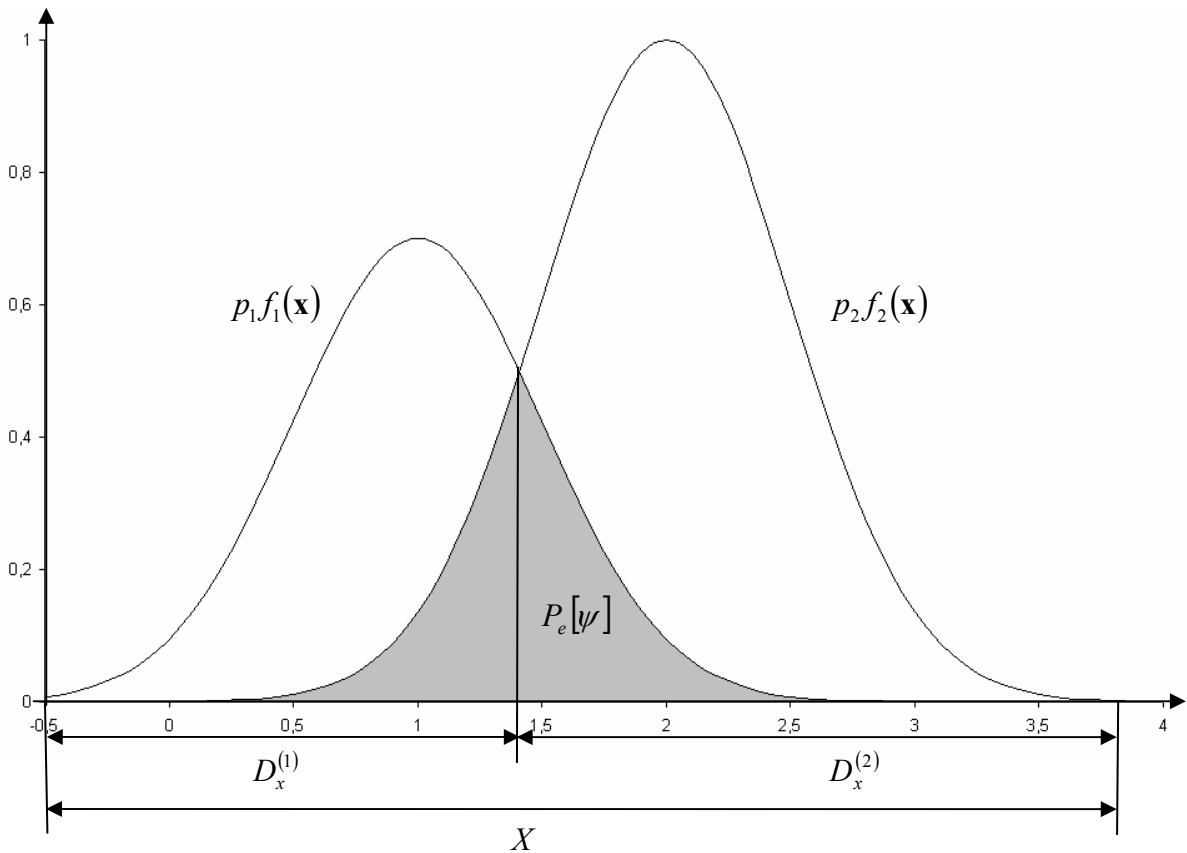
$$\psi^*(\mathbf{x}) = i, \text{ gdy } p_i f_i(\mathbf{x}) = \max_{k \in M} p_k f_k(\mathbf{x}). \quad (4.23)$$

Algorytm ten wybiera klasę, dla której występuje największe prawdopodobieństwo, że obiekt opisany wektorem cech \mathbf{x} do niej należy. Inaczej mówiąc, algorytm wybiera najbardziej prawdopodobną klasę dla wektora cech \mathbf{x} .

Na Rys. 4.5.2 przedstawiono przykład algorytmu klasyfikacji ψ dzielącego jednowymiarową przestrzeń cech na dwa obszary decyzyjne w sposób minimalizujący ryzyko błędnej klasyfikacji. Prawdopodobieństwo błędnej klasyfikacji zobrazowane jest przez zaciemniony obszar wykresu i wynosi

$$P_e[\psi] = \int_{D_x^{(1)}} p_2 f_2(x) dx + \int_{D_x^{(2)}} p_1 f_1(x) dx. \quad (4.24)$$

Prawdopodobieństwo $P_e[\psi]$ jest najmniejsze, gdy punkt rozdzielający spełnia równanie $p_1 f_1(x) = p_2 f_2(x)$. Każdy inny podział przestrzeni cech powoduje zwiększenie prawdopodobieństwa błędu.



Rys. 4.5.2. Podział przestrzeni cech i prawdopodobieństwo błędnej klasyfikacji dla klasyfikatora bayesowskiego

4.5.3. Klasyfikacja ze zbiorem uczącym

W praktyce pełna znajomość rozkładów zmiennych losowych \mathbf{X} i J jest rzadko znana. Istnieje jednak wiele metod estymacji prawdopodobieństw i gęstości (patrz [13]). Najczęściej wykorzystuje się w tym celu informacje zawarte w tzw. zbiorze uczącym, zawierającym obiekty opisane przez pary (\mathbf{x}_i, j_i) , gdzie \mathbf{x}_i jest wektorem cech, j_i numer klasy, do której należy obiekt \mathbf{x}_i . N elementów zbiór uczący ma postać

$$S = \{(\mathbf{x}_1, j_1), (\mathbf{x}_2, j_2), \dots, (\mathbf{x}_N, j_N)\}. \quad (4.25)$$

Zbiór S można podzielić na podzbiory utworzone z elementów należących do tej samej klasy. Podzbiór elementów należących do klasy j przyjmuje postać

$$S_j = \{(\mathbf{x}_k, j_k) : \mathbf{x}_k \in X, j_k = j, k = 1, 2, \dots, N_j\} \text{ dla } j \in M, \quad (4.26)$$

gdzie N_j jest liczbą obiektów klasy j w zbiorze uczącym S . Dla powyższych założeń zachodzi

$$S = \{S_1, S_2, \dots, S_M\}$$

oraz

$$N = \sum_{j=1}^M N_j. \quad (4.27)$$

Klasyfikacja ze zbiorem uczącym polega na oszacowaniu nieznanymi prawdopodobieństw i gęstości, a następnie użyciu ich w algorytmie optymalnym zamiast wartości rzeczywistych. Empiryczny algorytm bayesowski przyjmuje postać

$$\psi_S(\mathbf{x}) = i, \text{ gdy } \hat{p}_i \hat{f}_i(\mathbf{x}) = \max_{k \in M} \hat{p}_k \hat{f}_k(\mathbf{x}), \quad (4.28)$$

gdzie \hat{p}_i jest estymatorem prawdopodobieństwa *a priori* wyznaczanym najczęściej ze wzoru

$$\hat{p}_i = \frac{N_i}{N}, \quad (4.29)$$

natomiast $\hat{f}_i(\mathbf{x})$ jest estymatorem gęstości warunkowej $f_i(\mathbf{x})$. W analogiczny sposób można zastosować algorytm wykorzystujący bezpośrednio prawdopodobieństwo *a posteriori*. Algorytm taki przyjmuje postać

$$\psi_S(\mathbf{x}) = i, \text{ gdy } \hat{p}_i(\mathbf{x}) = \max_{k \in M} \hat{p}_k(\mathbf{x}). \quad (4.30)$$

gdzie $\hat{p}_i(\mathbf{x})$ jest estymatorem prawdopodobieństwa *a posteriori* $p_i(\mathbf{x})$.

Istnieje bardzo wiele metod estymacji gęstości warunkowej lub prawdopodobieństwa *a posteriori*, różniących się podejściem do problemu, złożonością obliczeniową, własnościami i ograniczeniami (patrz [6], [13]). Podstawową metodą statystyczną klasyfikacji ze zbiorem uczącym jest metoda k najbliższych sąsiadów (k -NN, z ang. k Nearest Neighbors). Zakłada ona, że jeśli wokół punktu \mathbf{x} zdefiniuje się pewien obszar, w który wpada dokładnie k obiektów ze zbioru uczącego S , przy czym k_i tych obiektów należy do klasy i -tej, to prawdopodobieństwo *a posteriori* można wyznaczyć (patrz [6]) korzystając ze wzoru

$$\hat{p}_i(\mathbf{x}) = \frac{k_i}{k}. \quad (4.31)$$

Algorytm ten zalicza obiekt do klasy najliczniej reprezentowanej w sąsiedztwie punktu \mathbf{x} i dla stałej liczby sąsiadów k przyjmuje postać

$$\psi_S^{(k-NN)}(\mathbf{x}) = i, \text{ gdy } k_i = \max_{j \in M} k_j. \quad (4.32)$$

Szczególnym przypadkiem metody k -NN, dla liczby sąsiadów k równej jeden, jest algorytm najbliższy sąsiad (NN), który zalicza badany obiekt do klasy, do której należy najbliższy obiekt w zbiorze uczącym.

Kolejnym dobrze znanym algorytmem jest algorytm najbliższa średnia (NM, z ang. Nearest Mean). Ideą algorytmu jest określenie dla każdej z klas średnich reprezentantów, a następnie wykorzystanie reguły NN do znalezienia najbliższego reprezentanta.

Istnieje szereg innych metod, które powstały na bazie inspiracji neuronowych, a nie bezpośrednio analizy statystycznej. Należy jednak pamiętać, że wszystkie one, w sposób bardziej lub mniej jawny, estymują gęstości lub prawdopodobieństwa. W przypadku klasyfikatorów neuronowych bardzo ważnym czynnikiem jest odpowiednie nauczanie klasyfikatora poprawnej klasyfikacji polegające na takim doborze wewnętrznych parametrów (wag sieci), aby zminimalizować błąd klasyfikacji.

W rozdziale 5 opisano wielowarstwowe sieci neuronowe, które można potraktować jako metodę regresji. W rozdziale 6 przedstawiono sieci LVQ należące do grupy klasyfikatorów, które do podziału przestrzeni cech wykorzystują reprezentantów klas.

4.5.4. Ocena jakości rozpoznawania

Każdy algorytm klasyfikacji może popełnić błąd. Na jakość klasyfikacji, czyli prawdopodobieństwo błędnej klasyfikacji, wpływa wiele czynników, do których należy np. dobór odpowiednich parametrów klasyfikatora, a dla klasyfikatorów neuronowych sposób, czas i parametry nauki. Oceny jakości klasyfikacji dokonuje się eksperymentalnie, co wymaga zastosowania dodatkowego zbioru przykładów, nazywanego zbiorem testującym. Zbiór testujący ma taką samą strukturę jak zbiór uczący, ale musi być od niego niezależny. Testowanie klasyfikatora polega na prezentacji przykładów ze zbioru testującego i wyznaczeniu liczby poprawnie i błędnie sklasyfikowanych przypadków.

W zależności od natury zadania rozpoznawania, utworzenie zbioru testującego może być trudne, co powodowane jest faktem, że wszystkie możliwe dane używane są w zbiorze uczącym i nie mogą być użyte w zbiorze testującym. W takich przypadkach stosuje się szereg metod oceny jakości klasyfikacji, do których należą metody wydzielania, usuwania i rotacji (opisane w [6]), polegających na odpowiednim podziale zbioru uczącego na dwa podzbiory, z których jeden służy do nauki, a drugi do testowania.

W przypadku badanego systemu problem ten nie występuje, gdyż dostęp do danych uczących jest łatwy i możliwe jest stworzenie niezależnych zbiorów uczących oraz testujących.

Ocena klasyfikacji może dotyczyć nie tylko ogólnego błędu klasyfikacji, ale także błędów związanych z myleniem klas. Może się zdarzyć, że klasyfikator daje dobry ogólny wynik rzędu 3% błędów, ale błąd ten jest powodowany całkowitym myleniem dwóch klas, podczas gdy pozostałe klasy rozróżniane są poprawnie. Aby mieć możliwość analizowania tego typu sytuacji należy wyniki klasyfikacji zebrać w tabeli, której przykład znajduje się poniżej.

	0	1	2	Brak decyzji	N	Do innych klas [%]
0	5	1	0	0	6	16
1	0	6	0	0	6	0
2	1	0	4	1	6	33
N	6	7	4	1	18	
Z innych klas [%]	16	14	0			28

Tabela 4.5.1. Przykład tabeli wyników klasyfikacji

Tabela 4.5.1 zawiera przykład wyników klasyfikacji dla trzech klas. Zbiór testujący składał się z 18 obiektów, przy czym każda z klas była reprezentowana przez 6 obiektów. Trzy pierwsze wiersze odpowiadają wynikom dla trzech klas. Analiza tabeli pozwala stwierdzić, że ogólny błąd klasyfikacji wyniósł 28%, 33% obiektów należących do klasy 2 zostało rozpoznanych jako obiekty innych klas, a najlepiej była rozpoznawana klasa 1 (wszystkie przypadki rozpoznane poprawnie). Ponadto można stwierdzić, że z obiektów zaklasyfikowanych do klasy zerowej 16% należy do innych klas.

5. Wielowarstwowa sieć jednokierunkowa

5.1. Wstęp

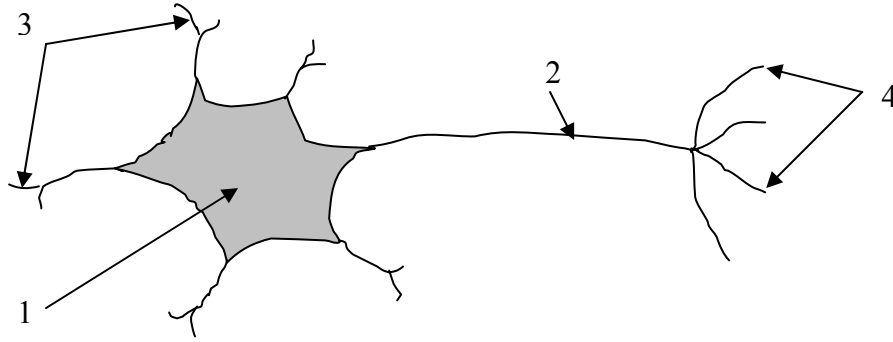
W dzisiejszych czasach sztuczne sieci neuronowe stanowią bardzo popularną i intensywnie rozwijającą się dziedzinę wiedzy, która znajduje zastosowanie w wielu obszarach nauki, a także życia codziennego. Sieć neuronowa składa się z połączonych ze sobą neuronów, których budowa i zasada działania wzorowana jest na zjawiskach zachodzących w mózgu człowieka. Sam neuron, jak i sieć, są bardzo uproszczonymi modelami struktur rzeczywistych, pomimo tego zachowują bardzo istotną cechę, którą jest zdolność uczenia się i uogólniania nabytej wiedzy.

Pierwszy matematyczny model komórki nerwowej został opracowany i opisany w 1943 roku przez McCullocha i Pittsa. W modelu tym wykorzystano ideę neuronu jako ważonego sumatora sygnałów wejściowych ze skokową funkcją aktywacji. Na przełomie lat pięćdziesiątych i sześćdziesiątych model ten został z powodzeniem wykorzystany przez Rosenblatta do budowy pierwszej jednokierunkowej sieci neuronowej zwanej perceptronem. W 1969 roku Minsky i Papert wykazali, że sieć jednowarstwowa typu perceptron ma duże ograniczenie, tj. nie jest w stanie nauczyć się odwzorowania tak fundamentalnej funkcji, jaką jest Ex-OR, a nie znano jeszcze skutecznych metod uczenia sieci wielowarstwowych. Z tego względu prace badawcze nad sieciami spowolniły się, a początkowy entuzjazm osłabł. Dopiero w połowie lat osiemdziesiątych, gdy McClelland i Rumelhart opracowali metodę uczenia poprzez wsteczną propagację błędów, ponownie nastąpił gwałtowny wzrost zainteresowania tą dziedziną. Obecnie sieci neuronowe pozwalają na dokonywanie odwzorowań bardzo złożonych funkcji. Dzięki nieliniowemu charakterowi mają znacznie większe możliwości modelowania niż wcześniej stosowane metody modelowania liniowego.

Sieci neuronowe znajdują zastosowanie w czterech głównych obszarach: aproksymacji, klasyfikacji, predykcji oraz asocjacji. W przypadku aproksymacji, sieć uczy się realizować zadaną nieliniową funkcję wielu zmiennych. Zadanie klasyfikacji polega na przyporządkowaniu numeru klasy do wzorca podanego na wejście sieci, przy czym sieć uczy się wykorzystywać informacje o cechach charakterystycznych dla danych klas. W przypadku predykcji sieć neuronowa przewiduje przyszłe wartości w ciągu próbek, na podstawie wartości z przeszłości. W zadaniu asocjacji sieć neuronowa pełni rolę pamięci asocjacyjnej i ma zdolność uzupełniania wzorców wejściowych o brakujące (np. uszkodzone) elementy.

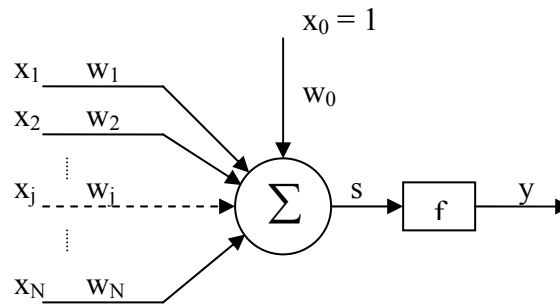
5.2. Neuron

Podstawowym elementem systemu nerwowego jest komórka neuronowa nazywana neuronem. Neuron, przedstawiony na Rys. 5.2.1, składa się z ciała komórki, wypustek wprowadzających informacje do neuronu nazywanych dendrytami oraz wypustki wyprowadzającej nazywanej aksonem, która poprzez synapsy przekazuje informacje do innych neuronów. Każdy neuron może mieć wiele wejść, ale ma tylko jedno wyjście. Średnio neuron posiada kilka tysięcy synaps, czyli połączeń z innymi neuronami. Do neuronu docierają sygnały, z których każdy może wywierać hamujący bądź pobudzający wpływ. Aktywacja neuronu zależy od stanu wejść, jednak związek między stanem wejść, a stanem wyjścia może być bardzo złożony. Transmisja sygnałów odbywa się na drodze skomplikowanych procesów chemiczno-elektrycznych. Pomiedzy bodźcem wejściowym, a odpowiedzią neuronu może wystąpić znaczne opóźnienie. Działanie neuronu może zależeć od zmęczenia i zmieniać się w czasie pod wpływem nawet przypadkowych zdarzeń.



Rys. 5.2.1. Schemat neuronu: 1 - ciało komórki, 2 - akson, 3 - dendryty, 4 – synapsy

Po znacznych uproszczeniach przedstawiony powyżej neuron można zamodelować za pomocą sumatora ważonego z nieliniową funkcją przejścia. Typowy, często stosowany model neuronu przedstawiony jest na Rys. 5.2.2.



Rys. 5.2.2. Model neuronu

Neuron posiada N wejść oznaczonych przez $x_1 \dots x_N$. Dodatkowo wyposażony jest w wejście x_0 , dla którego przyjmuje się stan aktywacji zawsze równy 1. Z każdym wejściem związana jest waga synaptyczna: $w_0 \dots w_N$. Wagi mogą mieć wartości zarówno dodatnie (wejście pobudzające), jak i ujemne (wejście hamujące). Sygnał wyjściowy neuronu określone jest wzorem

$$y = f(s), \quad (5.1)$$

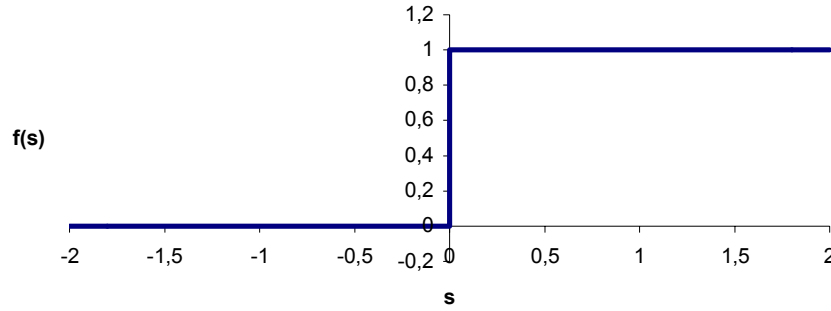
gdzie

$$s = \sum_{i=0}^N w_i x_i. \quad (5.2)$$

Funkcja progowa f może przyjmować różną postać. W modelu McCullocha i Pittsa jest to funkcja skoku jednostkowego wyrażona wzorem

$$f(s) = \begin{cases} 1 & \text{gdy } s \geq 0 \\ 0 & \text{gdy } s < 0 \end{cases}, \quad (5.3)$$

której postać przedstawia Rys. 5.2.3.



Rys. 5.2.3. Wykres skokowej funkcji aktywacji

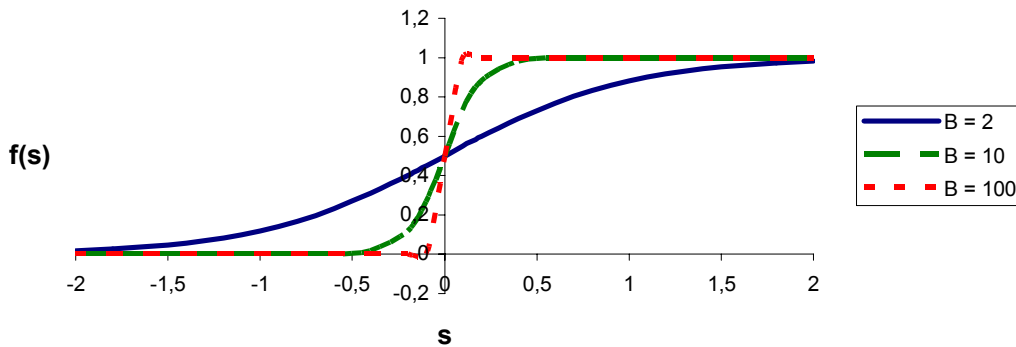
Możliwe jest także wykorzystanie bipolarnej funkcji skokowej, dla której wartość sygnału wyjściowego jest równa ± 1 . Do uczenia neuronów ze skokową funkcją aktywacji można zastosować regułę Widrowa-Hoffa opisaną m.in. w [1].

Wadą skokowych funkcji aktywacji jest nieróżniczkowalność, która uniemożliwia ich wykorzystanie w przypadku gradientowych metod uczenia, wymagających różniczkowalnej funkcji błędu. Obecnie najczęściej stosuje się sigmoidalne funkcje aktywacji, stanowiące ciągle przybliżenie funkcji skokowych, których dodatkową zaletą jest łatwość obliczanie pochodnej $f'(s)$ na podstawie wartości funkcji $f(s)$. W wersji unipolarnej funkcja taka przyjmuje dowolne wartości z przedziału $(0, 1)$ i najczęściej stosuje się tzw. funkcję logistyczną definiowaną jako

$$f(s) = \frac{1}{1 + e^{-\beta s}}, \quad (5.4)$$

gdzie $\beta > 0$ jest parametrem. Pochodna funkcji logistycznej wynosi

$$f'(s) = \beta f(s)(1 - f(s)). \quad (5.5)$$

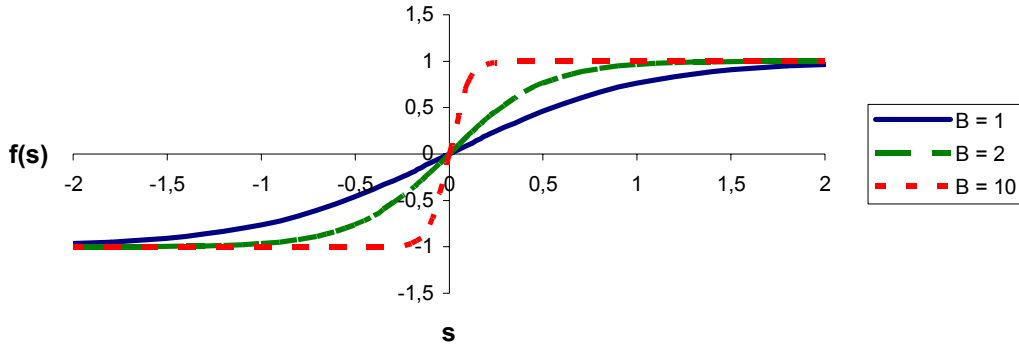
Rys. 5.2.4. Wykres logistycznej funkcji aktywacji dla różnych wartości parametru β

W wersji bipolarnej funkcja przyjmuje wartości z przedziału $(-1, 1)$ i najczęściej używa się funkcji tangensa hiperbolicznego

$$f(s) = \tanh(\beta s), \quad \text{gdzie } \beta > 0, \quad (5.6)$$

którego pochodną można łatwo obliczyć ze wzoru

$$f'(s) = \beta(1 - f^2(s)). \quad (5.7)$$

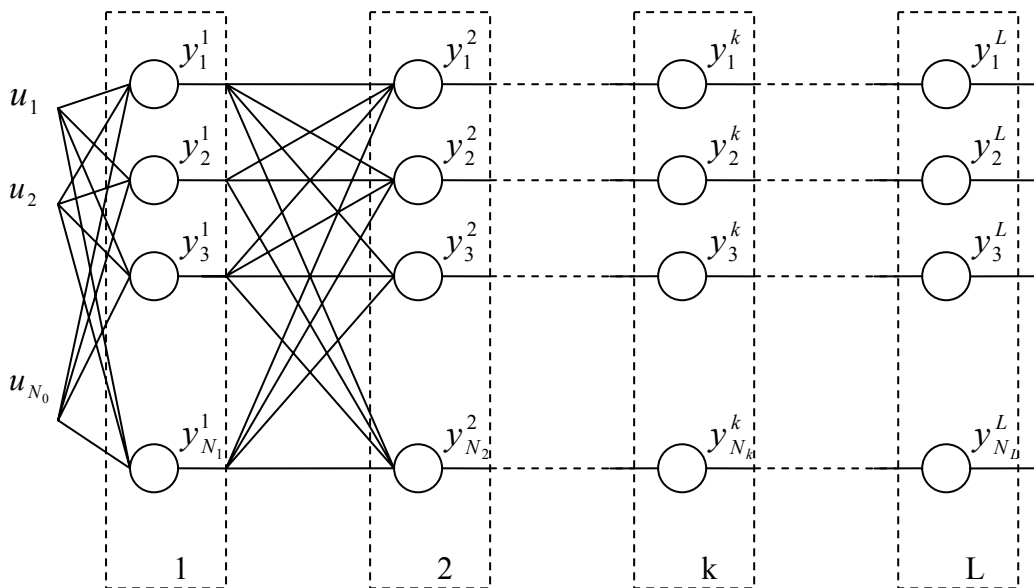


Rys. 5.2.5. Wykres tangensoidalnej funkcji aktywacji dla różnych wartości parametru β

Należy zauważyć, że w obu przypadkach, gdy $\beta \rightarrow \infty$, to funkcje są zbieżne do funkcji progowych. Im większy jest współczynnik β , tym bardziej stroma jest postać funkcji, co dla funkcji logistycznej można zaobserwować na Rys. 5.2.4, a dla funkcji tangensa hiperbolicznego na Rys. 5.2.5.

5.3. Struktura sieci MLP

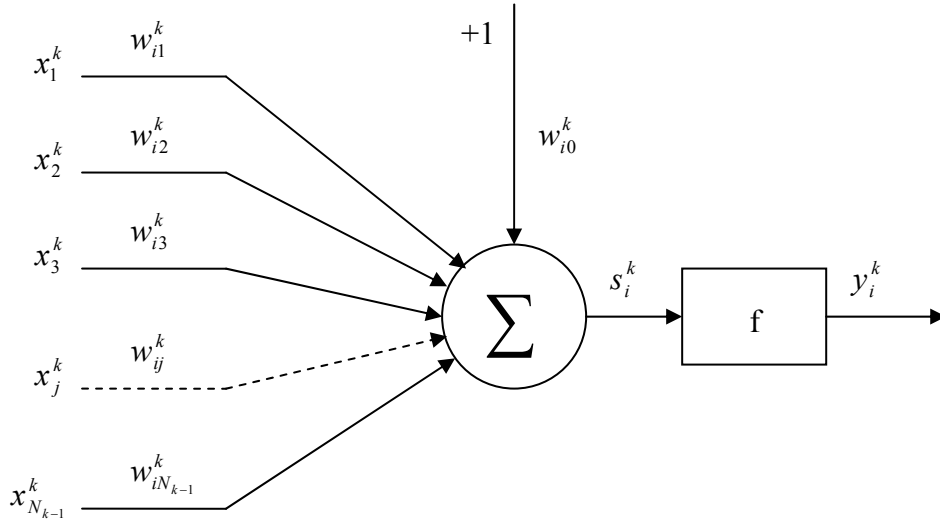
Typowa jednokierunkowa sieć neuronowa MLP (z ang. Multi Layer Perceptron) składa się z L warstw. Każda warstwa zawiera N_k neuronów, gdzie $k = 1, \dots, L$ jest numerem warstwy. Sieć ma N_0 wejść i N_L wyjść. W chwili n na wejście sieci podawane są sygnały $u_i(n)$, gdzie $i = 1, \dots, N_0$. Sygnał wyjściowy i -tego neuronu k -tej warstwy jest oznaczony przez $y_i^k(n)$, $i = 1, \dots, N_k$, $k = 1, \dots, L$. Strukturę sieci przedstawia Rys. 5.3.1. Należy dodać, że niektórzy autorzy traktują wejścia sieci jako oddzielną warstwę, która nie ma jednak zdolności przetwarzających. Przy takim podejściu stwierdzenie „sieć trójwarstwowa” oznacza, że sieć ma warstwę wejściową (bez zdolności obliczeniowych), jedną warstwę ukrytą oraz warstwę wyjściową.



Rys. 5.3.1. Struktura sieci MLP

Neuron i -ty warstwy k -tej posiada $N_{k-1} + 1$ wejść: $x_i^k(n)$, $i = 0, \dots, N_{k-1}$. Przy czym $x_0^k(n) = 1$. Każde z wejść warstwy k -tej jest powiązane z wyjściem warstwy wcześniejszej (warstwy $k-1$) w następujący sposób

$$x_i^k = \begin{cases} u_i(n) & k = 1 \\ y_i^{k-1}(n) & k = 2, \dots, L \\ 1 & i = 0, k = 1, \dots, L \end{cases} \quad (5.8)$$



Rys. 5.3.2. Schemat neuronu w sieci wielowarstwowej

Na powyższym rysunku przez $w_{ij}^k(n)$ oznaczono wagę i -tego neuronu, gdzie $i = 1, \dots, N_k$, k -tej warstwy, łączącą ten neuron z j -tym sygnałem wejściowym $x_j^k(n)$, $j = 0, \dots, N_{k-1}$.

Sygnał wyjściowy neuronu i -tego w warstwie k -tej jest określany wzorem

$$y_i^k(n) = f(s_i^k(n)), \quad (5.9)$$

gdzie

$$s_i^k(n) = \sum_{j=0}^{N_{k-1}} w_{ij}^k(n) x_j^k(n). \quad (5.10)$$

Sygnał wyjściowy warstwy L -tej $y_1^L(n) \dots y_{N_L}^L(n)$ jest jednocześnie sygnałem wyjściowym całej sieci neuronowej.

5.4. **Uczenie sieci – algorytm wstecznej propagacji błędów**

Uczenie sieci neuronowej ma na celu takie określenie wartości wag wszystkich neuronów sieci, aby przy zadanym wektorze wejściowym $u(n)$ uzyskać na wyjściu sieci wartości $y^L(n)$ jak najbliższe wartościom żądanym $d(n)$. Ten typ uczenia, gdzie prezentuje się sieci wektor wejściowy oraz związany z nim pożądaný wektor wyjściowy, nazywa się w literaturze uczeniem z nauczycielem. Jeśli podamy na wejście sieci sygnał $u_i(n)$, gdzie $i = 1, \dots, N_0$, to po przetworzeniu otrzymamy na wyjściu sygnał $y_1^L(n) \dots y_{N_L}^L(n)$, który

porównywany jest z sygnałem żądanym $d_1(n) \dots d_{N_L}(n)$, w wyniku czego otrzymujemy błąd dla ostatniej warstwy

$$\varepsilon_i^L(n) = d_i(n) - y_i^L(n), \quad i = 1, \dots, N_L. \quad (5.11)$$

Jeżeli przez $Q(n)$ oznaczymy pewną miarę błędu, która jest funkcją wag sieci, to uczenie sieci polega na korekcji wszystkich wag w taki sposób, aby zminimalizować tę miarę. Do korekcji wag można użyć reguły najszybszego spadku, mającej następującą postać

$$w_{ij}^k(n+1) = w_{ij}^k(n) + \Delta w_{ij}^k(n), \quad (5.12)$$

gdzie

$$\Delta w_{ij}^k(n) = -\eta \frac{\partial Q(n)}{\partial w_{ij}^k(n)}, \quad (5.13)$$

Występująca powyżej stała $\eta > 0$ jest współczynnik uczenia określającym wielkość kroku. Można zauważyć, że po przekształceniu ostatniej części otrzymamy:

$$\frac{\partial Q(n)}{\partial w_{ij}^k(n)} = \frac{\partial Q(n)}{\partial s_i^k(n)} \frac{\partial s_i^k(n)}{\partial w_{ij}^k(n)} = \frac{\partial Q(n)}{\partial s_i^k(n)} x_j^k(n). \quad (5.14)$$

Po oznaczeniu

$$\delta_i^k(n) = \frac{\partial Q(n)}{\partial s_i^k(n)} \quad (5.15)$$

otrzymujemy

$$\frac{\partial Q(n)}{\partial w_{ij}^k(n)} = \delta_i^k(n) x_j^k(n). \quad (5.16)$$

Stąd algorytm uczenia przyjmuje postać

$$w_{ij}^k(n+1) = w_{ij}^k(n) - \eta \delta_i^k(n) x_j^k(n). \quad (5.17)$$

Sposób obliczania $\delta_i^k(n)$ zależy od przyjętej miary błędu oraz numeru warstwy, dla której jest liczony. W przypadku warstwy wyjściowej obliczenia te można zrobić bezpośrednio, lecz w przypadku warstw ukrytych zadanie to nie jest już takie proste i wymaga użycia specjalnego sposobu postępowania zwanego wsteczną propagacją.

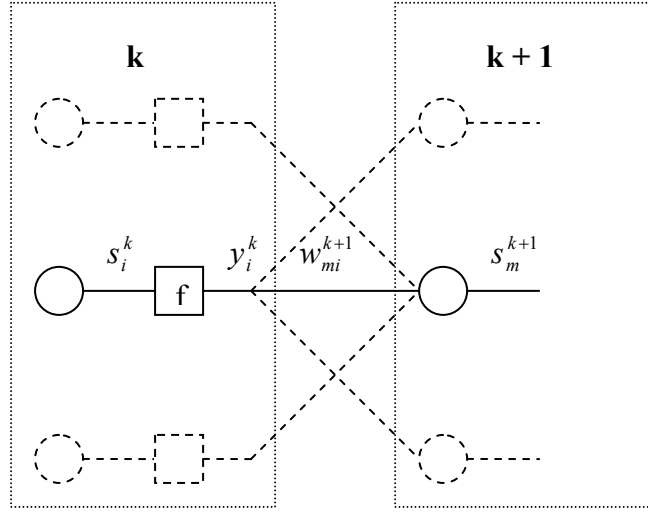
Klasyczną miarą błędu jest suma kwadratów różnic sygnału wyjściowego i żądanego, tzn.

$$Q(n) = \frac{1}{2} \sum_{i=1}^{N_L} \varepsilon_i^L(n)^2 = \frac{1}{2} \sum_{i=1}^{N_L} (y_i^L(n) - d_i(n))^2. \quad (5.18)$$

Dla warstwy wyjściowej $k = L$ wyrażenie $\delta_i^L(n)$ przyjmuje postać

$$\begin{aligned} \delta_i^L(n) &= \frac{\partial Q(n)}{\partial s_i^L(n)} = \frac{\partial \frac{1}{2} \sum_{m=1}^{N_L} \varepsilon_m^L(n)^2}{\partial s_i^L(n)} = \frac{1}{2} \frac{\partial \varepsilon_i^L(n)^2}{\partial s_i^L(n)} = \varepsilon_i^L(n) \frac{\partial \varepsilon_i^L(n)}{\partial s_i^L(n)} = \\ &= \varepsilon_i^L(n) \frac{\partial y_i^L(n)}{\partial s_i^L(n)} = \varepsilon_i^L(n) f'(s_i^L(n)). \end{aligned} \quad (5.19)$$

Dla dowolnej warstwy ukrytej ($k \neq L$), korzystając z faktu, że



$$s_m^{k+1}(n) = \sum_{j=0}^{N_k} w_{mj}^{k+1}(n) y_j^k(n) = \sum_{j=0}^{N_k} w_{mj}^{k+1}(n) f(s_j^k(n)) \quad (5.20)$$

oraz

$$\frac{\partial s_m^{k+1}(n)}{\partial s_i^k(n)} = w_{mi}^{k+1}(n) f'(s_i^k(n)), \quad (5.21)$$

otrzymujemy

$$\begin{aligned} \delta_i^k(n) &= \frac{\partial Q(n)}{\partial s_i^k(n)} = \sum_{m=1}^{N_{k+1}} \frac{\partial Q(n)}{\partial s_m^{k+1}(n)} \frac{\partial s_m^{k+1}(n)}{\partial s_i^k(n)} = \sum_{m=1}^{N_{k+1}} \delta_m^{k+1}(n) w_{mi}^{k+1}(n) f'(s_i^k(n)) = \\ &= f'(s_i^k(n)) \sum_{m=1}^{N_{k+1}} \delta_m^{k+1}(n) w_{mi}^{k+1}(n). \end{aligned} \quad (5.22)$$

Jeśli oznaczymy błąd dla warstwy k-tej wzorem

$$\varepsilon_i^k(n) = \sum_{m=1}^{N_{k+1}} \delta_m^{k+1}(n) w_{mi}^{k+1}(n) \text{ dla } k = 1, \dots, L-1, \quad (5.23)$$

otrzymamy

$$\delta_i^k(n) = \varepsilon_i^k(n) f'(s_i^k(n)). \quad (5.24)$$

Jak widać powyżej, błąd w warstwie k-tej obliczany jest na podstawie błędów w warstwie następnej, stąd nazwa algorytmu.

Podsumowując, ogólny wzór umożliwiający obliczenie poprawek dla wag w dowolnej warstwie według klasycznego algorytmu wstecznej propagacji błędów przyjmuje postać

$$w_{ij}^k(n+1) = w_{ij}^k(n) - \eta \delta_i^k(n) x_j^k(n), \quad (5.25)$$

gdzie

$$\delta_i^k(n) = \varepsilon_i^k(n) f'(s_i^k(n)) \quad (5.26)$$

oraz

$$\varepsilon_i^k(n) = \begin{cases} y_i^L(n) - d_i(n) & \text{dla } k = L \\ \sum_{m=1}^{N_{k+1}} \delta_m^{k+1}(n) w_{mi}^{k+1}(n) & \text{dla } k = 1, \dots, L-1 \end{cases} \quad (5.27)$$

Algorytm wstecznej propagacji błędów jest algorytmem wolno zbieżnym, do obliczeń wykorzystuje jedynie gradient funkcji błędu, a nie bierze pod uwagę np. hesjanu, tj. macierzy drugich pochodnych $Q(n)$ po wagach sieci, który niesie informacje o krzywiznie funkcji Q . Ponadto przedstawiony algorytm jest szczególnie mało efektywny w obszarach gdzie gradient przyjmuje małą wartość (np. w okolicach minimum). Pomimo swych wad, jest to jednak jeden z najczęściej używanych algorytmów uczenia wielowarstwowych sieci neuronowych, co jest powodowane jego prostotą w implementacji i stosunkowo małą złożonością obliczeniową.

Należy zaznaczyć, że algorytm ten jest algorytmem optymalizacji lokalnej. Funkcja błędu ma najczęściej bardzo dużą liczbę minimów lokalnych i powyższy algorytm może kończyć pracę w różnych minimach w zależności od punktu startowego, czyli wartości wag początkowych. Z tego względu zaleca się startowanie algorytmu z kilku losowo wybranych punktów początkowych i wybranie najlepszego rezultatu.

5.5. Sieć wielowarstwowa jako klasyfikator

Zastosowanie wielowarstwowej sieci neuronowej jako klasyfikatora wymaga specjalnego przygotowania zbioru uczącego oraz określenia metody interpretacji stanu wyjść sieci. Wyjście sieci neuronowej jest pewnym wektorem liczb rzeczywistych, a od klasyfikatora oczekujemy numeru klasy, do której przyporządkowuje on podany na wejście wektor cech. Najskuteczniejszą metodą podejścia do tego problemu jest ustalenie, że liczba wyjść sieci równa jest liczbie klas, do których sieć ma klasyfikować. Jednym ze sposobów postępowania jest przyjęcie, że sieć zaklasyfikowała wejściowy wektor cech do klasy i -tej, jeśli wyjście i -te jako jedyne znajduje się w stanie aktywacji powyżej pewnego ustalonego progu. Inna metoda zakłada, że wyjście i -te musi być w stanie aktywacji przekraczającym o pewną wartość wszystkie pozostałe wyjścia. W pracy tej przyjęto pierwszą metodę z progiem równym 0,5. W takim przypadku, za wynik klasyfikacji przyjmuje się klasę i -tą, jeżeli

$$y_i^L(n) > 0,5 \text{ oraz } y_j^L(n) \leq 0,5 \text{ dla każdego } j \neq i. \quad (5.28)$$

Jak łatwo zauważyć, możliwe są sytuacje, w których aktywacja żadnego z wyjść nie będzie przekraczała ustalonego progu, lub aktywacja większej liczby neuronów będzie ten próg przekraczała. W obu przypadkach przyjmuje się, że sieć neuronowa nie jest w stanie zaklasyfikować wektora cech podanego na wejście.

Drugim ważnym zagadnieniem, które pojawia się, gdy sieć neuronowa ma być użyta jako klasyfikator, jest odpowiednia konstrukcja zbioru uczącego, a konkretnie żądanych wektorów wyjściowych używanych przy uczeniu. Jak wspomniano w punkcie 4.5, zbiór uczący składa się z par (\mathbf{x}, c) , gdzie: \mathbf{x} – wektor cech, c – numer (etykieta) klasy. Jeśli założymy podaną wyżej interpretację wyjścia sieci, to należy stworzyć wektory wyjść żądanych \mathbf{d} na podstawie numeru klasy w taki sposób, aby składnik wektora odpowiadający poprawnej klasie miał wartość największą, a pozostałe składniki wartości minimalne. Należy zauważyć, że nie jest zalecane przyjmowanie w wektorze \mathbf{d} ekstremalnych wartości aktywacji (np. 0 i 1 dla funkcji logistycznej), gdyż i tak nie będą one nigdy osiągnięte, a przy uczeniu może pojawić się numeryczna niestabilność związana z podciąganiem wag do bardzo dużych wartości. Ponadto pochodna zarówno funkcji logistycznej, jak i funkcji tangensa hiperbolicznego dla wartości ekstremalnych dąży do zera, więc uczenie metodami

gradientowymi może być bardzo powolne. Z tego względu zaleca się uczenie sieci osiągania wartości umiarkowanych. W przypadku logistycznej funkcji aktywacji przyjmuje się, że jeżeli żadaną klasą jest klasa c , to wektor \mathbf{d} ma składowe równe

$$d_i = \begin{cases} 0.9 & \text{gdy } i = c \\ 0.1 & \text{w przeciwnym przypadku} \end{cases} \quad (5.29)$$

Dla bipolarnej funkcji aktywacji przyjmuje się najczęściej

$$d_i = \begin{cases} 0.9 & \text{gdy } i = c \\ -0.9 & \text{w przeciwnym przypadku} \end{cases} \quad (5.30)$$

5.6. Momentum

Jak zauważono w punkcie 5.4, algorytm wstecznej propagacji błędów w swojej podstawowej formie jest wolno zbieżny, szczególnie w płaskich odcinakach funkcji błędu, gdy gradient ma małe wartości. Poprawę efektywności można uzyskać wykorzystując przy uczeniu tzw. *momentum*. Momentum jest dodatkowym składnikiem brany pod uwagę przy aktualizacji wartości wag. Składnik ten równy jest ostatniemu przyrostowi danej wagi pomnożonemu przez pewną stałą α o wartości z przedziału $[0,1]$. Algorytm uaktualniania wag przyjmuje postać

$$w_{ij}^k(n+1) = w_{ij}^k(n) + \Delta w_{ij}^k(n), \quad (5.31)$$

gdzie

$$\Delta w_{ij}^k(n) = -\eta \frac{\partial Q(n)}{\partial w_{ij}^k(n)} + \alpha(w_{ij}^k(n) - w_{ij}^k(n-1)). \quad (5.32)$$

Składnik *momentum* jest niezależny od aktualnej wartości gradientu, zależy jedynie od ostatniej zmiany wagi. Jego wpływ rośnie na płaskich odcinkach funkcji celu, powodując przyspieszenie procesu nauki.

W pobliżu minimum, gdzie gradient przyjmuje małe wartości, *momentum* jako składnik dominujący może spowodować zmianę wartości wag w taki sposób, że minimum funkcji błędu zostanie „przeskoczony”, a wartość tej funkcji wzrośnie. Aby zapobiec takim przypadkom sprawdza się wartość funkcji celu po wykonaniu kroku, jeśli jest ona większa od wartości w poprzednim kroku o zadaną wartość, określoną przez współczynnik dopuszczalnego wzrostu błędu (typowo 1,05), to krok zostaje cofnięty i przyjmuje się

$$w_{ij}^k(n) - w_{ij}^k(n-1) = 0. \quad (5.33)$$

W ten sposób składnik związany z gradientem staje się składnikiem dominującym i proces uczenia ponownie przebiega zgodnie z kierunkiem gradientu.

5.7. Dobór współczynnika uczenia

Odpowiedni dobór wielkości współczynnika uczenia ma zasadniczy wpływ na szybkość i zbieżność procesu uczenia. Przy zbyt małym współczynniku wykonywane przez algorytm kroki będą bardzo małe i aby osiągnąć minimum potrzebna będzie duża liczba iteracji. Natomiast w przypadku zbyt dużej wartości współczynnika uczenia, zbyt duże kroki mogą powodować „przeskoczenie” minimum, co może prowadzić do braku zbieżności.

5.7.1. Stały współczynnik uczenia – MLP1

Najprostsza, a zarazem najmniej efektywna metodą doboru współczynnika uczenia polega na przyjęciu jego stałej wartości. W literaturze (np. [1]) sugeruje się, że w takim przypadku każda warstwa powinna mieć oddzielnie dobrany współczynnik o wartości nie większej niż $1/n$, gdzie n jest liczbą wejść warstwy. W niniejszej pracy zastosowano jednak prostszą metodą, w której współczynnik uczenia jest taki sam dla wszystkich warstw. Na potrzeby pracy metodę tę, wraz z zastosowaniem momentum, oznaczono jako MLP1.

5.7.2. Dobór współczynnika na podstawie wyników klasyfikacji – MLP2

Lepszym rozwiązaniem jest dobieranie współczynnika uczenia w zależności od rezultatów uzyskiwanych przez sieć. W pracy [2] zaproponowano dobór wielkości współczynnika, dla sieci używanej do klasyfikacji, na podstawie liczby poprawnie rozpoznawanych wzorców ze zbioru uczącego. W metodzie tej autorzy nie używali *momentum*. Współczynnik uczenia oblicza się według zależności

$$\eta = \alpha \cdot A + \beta, \quad (5.34)$$

gdzie

A – procentowy stosunek poprawnie rozpoznanych wzorców zbioru uczącego do ilości wzorców w tym zbiorze

$$\alpha = \begin{cases} (\eta_{100} - \eta_{95})/5 & \text{gdy } A > 95 \\ (\eta_{95} - \eta_0)/95 & \text{w przeciwnym przypadku} \end{cases},$$

$$\beta = \begin{cases} 20 \cdot \eta_{95} - 19 \cdot \eta_{100} & \text{gdy } A > 95 \\ \eta_0 & \text{w przeciwnym przypadku} \end{cases}, \quad (5.35)$$

$\eta_0, \eta_{95}, \eta_{100}$ - stałe współczynniki uczenia dla A równego: 0, 95 i 100%

Taki dobór współczynnika uczenia umożliwia przyspieszenie uczenia w przypadku małych osiągnięć, tj. dla dużych błędów rozpoznawania, oraz zwolnienie w przypadku dużych osiągnięć, aby zapobiec przeskoczeniu minimum.

W dalszej części pracy metoda ta nazywana jest MLP2.

5.7.3. Adaptacyjny dobór współczynnika uczenia – MLP3

Jeszcze inną strategią zmian wartości współczynnika uczenia jest dobór na podstawie zmian wartości funkcji błędu w trakcie procesu uczenia. Jeżeli przez $\varepsilon(n)$ oraz $\varepsilon(n-1)$ oznaczymy średni błąd po całym zbiorze uczącym w iteracjach odpowiednio n -tej i $(n-1)$ -szej, a przez $\eta(n)$ współczynnik uczenia w iteracji n -tej, to wartość współczynnika uczenia w kolejnej iteracji wyznaczamy z zależności

$$\eta(n+1) = \begin{cases} \eta(n)p_i & \text{gdy } \varepsilon(n) > k\varepsilon(n-1) \\ \eta(n)p_d & \text{gdy } \varepsilon(n) \leq k\varepsilon(n-1) \end{cases}, \quad (5.36)$$

gdzie przez k oznaczono współczynnik dopuszczalnego wzrostu błędu, p_i – współczynnik zwiększania wartości η , p_d – współczynnik zmniejszania wartości η . Oznacza to, że współczynnik uczenia rośnie, jeśli błąd maleje, co powoduje przyspieszenie procesu nauki. W

przeciwnym przypadku, gdy błąd zaczyna rosnąć powyżej dopuszczalnej wartości, współczynnik uczenia maleje, a krok zostaje cofnięty.

Odpowiedni dobór powyższych parametrów zależy od konkretnego problemu (postaci funkcji błędu) i jest kluczowym czynnikiem wpływającym na efektywność algorytmu. W pracy [1] proponowane są wartości $k = 1,04$, $p_i = 1,05$, $p_d = 0,7$.

5.7.4. Inne metody doboru współczynnika uczenia

Istnieje szereg innych metod doboru współczynnika uczenia, z których ze względu na efektywność wyróżnić należy dobór poprzez minimalizację kierunkową. Metoda ta polega na znajdowaniu minimum funkcji błędu na kierunku wyznaczonym za pomocą gradientu, tzn. dobranie takiej wielkości kroku, aby po jego wykonaniu przejść do najlepszego (z pewnym przybliżeniem) rozwiązania w tym kierunku. Metoda jest bardzo kosztowna, jeśli chodzi o wyznaczanie współczynnika uczenia, ale powoduje znaczne zredukowanie liczby iteracji potrzebnych do osiągnięcia minimum. Ze względu na to, że metoda ta nie jest wykorzystywana w pracy, nie będą tu podawane jej szczegóły (opisane w [1]).

5.8. Inne metody uczenia sieci wielowarstwowych

W punkcie 5.4 przedstawiono podstawowy algorytm uczenia wielowarstwowych sieci neuronowych poprzez wsteczną propagację błędów. Jak zaznaczono, algorytm ten jest wolno zbieżnym algorytmem optymalizacji lokalnej. Istnieje szereg bardziej zaawansowanych algorytmów nauki sieci, które nie będą tu w szczególności opisywane, gdyż nie zostały wykorzystane w pracy.

Bardziej złożone algorytmy przy obliczaniu poprawek na wagi biorą pod uwagę nie tylko gradient funkcji błędu, ale także hesjan tej funkcji lub aproksymację hesjanu (np. w metodzie Levenberga-Marquardta, patrz [1]). Należy jednak zauważyć, że zastosowanie tego typu algorytmów nie polepsza jakości wyniku, minimalizacja nadal kończy się w jednym z minimów lokalnych, ale osiągnięcie tego minimum może nastąpić w znacznie krótszym czasie.

Inną grupę algorytmów stanowią algorytmy optymalizacji globalnej, do których zaliczyć można m.in. algorytmy genetyczne i symulowane wyżarzanie. Algorytmy te przeszukują przestrzeń rozwiązań w celu znalezienia globalnego minimum funkcji błędu poprzez losowe zaburzenia wartości wektorów wag. Zaburzenie takie może spowodować opuszczenie minimum lokalnego i przejście w inne regiony przestrzeni rozwiązań. Możliwe jest także połączenie metod optymalizacji globalnej z metodami gradientowymi, np. zastosowanie algorytmu genetycznego do znajdowania wag i algorytmu wstecznej propagacji, odnajdującego najbliższe minimum lokalne. Metody optymalizacji globalnej są zazwyczaj bardzo kosztowne obliczeniowo, ale umożliwiają jakościowe poprawienie wyników poprzez ucieczkę z minimów lokalnych.

6. Sieć LVQ

6.1. Wstęp

Innym rodzajem sieci neuronowych, które w ostatnich latach zyskały na znaczeniu, są sieci samoorganizujące. W odróżnieniu od sieci MLP, sieci samoorganizujące najczęściej trenowane są metodą *bez nauczyciela (bez nadzoru)*, oznacza to, że sygnałom uczącym nie towarzyszą sygnały żądane na wyjściu sieci, a sieć sama uczy się wykrywać istotne cechy sygnałów wejściowych. Wyróżnia się dwie główne grupy tego typu sieci: działające na podstawie reguły Hebba oraz działające na zasadzie współzawodnictwa. W przypadku sieci wykorzystujących regułę Hebba, zmiana wagi uzależniona jest od stanu aktywacji neuronów połączonych tą wagą. Im większe pobudzenie, tym większy przyrost wagi, co wprowadza pewnego rodzaju dodatnie sprzężenie zwrotne. W przypadku sieci działających na zasadzie współzawodnictwa, wykorzystuje się konkurencję między neuronami do wyłonienia neuronu o największej aktywacji. Neuron wygrywający i ewentualnie jego sąsiedzi dopasowują swoje wagi do sygnału wejściowego. W obu przypadkach, podczas uczenia można zaobserwować wzrost globalnego uporządkowania sieci.

Sieci samoorganizujące typu konkurencyjnego były szczegółowo badane w latach osiemdziesiątych przez Kohonena, który zaproponował szereg skutecznych algorytmów uczenia tego typu sieci. W roku 1986 Kohonen zaproponował bardzo efektywną metodę rozpoznawania wzorców, która nosi nazwę adaptacyjnego kwantowania wektorowego – LVQ (z ang. Learning Vector Quantization). Metoda ta bazuje na idei działania sieci samoorganizującej, ale podczas uczenia wykorzystuje dodatkowo informacje o żądanym wyjściu (numerze klasy).

6.2. Sieci samoorganizujące Kohonena

Zasada działania sieci samoorganizujących Kohonena (SOM – Self-Organizing Maps) jest odmienna, niż sieci wielowarstwowych MLP opisanych w poprzednim rozdziale. Sieci Kohonena składają się zazwyczaj z jednej warstwy zawierającej k neuronów. Każdy z neuronów warstwy jest połączony ze wszystkimi składowymi wektora wejściowego $\mathbf{x} \in R^n$ poprzez wagi $\mathbf{m}_i \in R^n$, gdzie $i = 1, 2, \dots, k$. We współzawodnictwie wygrywa ten neuron, którego wektor wag jest najbliższy wektorowi wejściowemu w sensie przyjętej metryki. Oznacza to, że zwycięża neuron c , jeżeli spełniona jest zależność

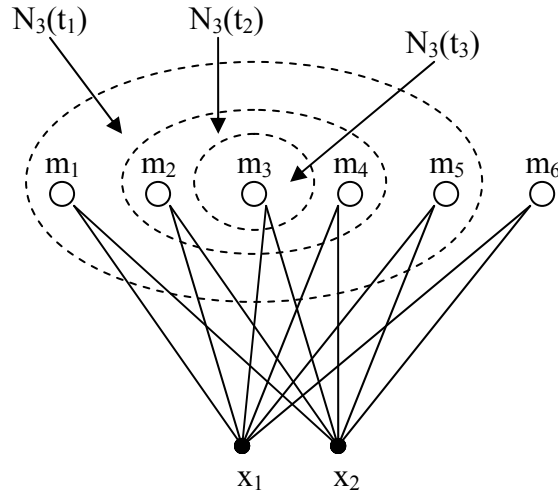
$$d(\mathbf{x}, \mathbf{m}_c) = \min_{1 \leq i \leq k} d(\mathbf{x}, \mathbf{m}_i), \quad (6.1)$$

gdzie $d(\mathbf{x}, \mathbf{m})$ oznacza odległość dla przyjętej metryki między wektorami \mathbf{x} i \mathbf{m} . Najczęściej używa się metryki euklidesowej, w takim przypadku odległość między wektorami oblicza się korzystając ze wzoru

$$d(\mathbf{x}, \mathbf{m}) = \|\mathbf{x} - \mathbf{m}\| = \sqrt{\sum_{j=1}^n (x_j - m_j)^2} \quad (6.2)$$

Dodatkowo ustala się topologię sieci, przy czym w tym wypadku nie odzwierciedla ona sposobu połączenia neuronów z wejściem i nie jest związane z wymiarem wektora wejściowego. Topologia określa jedynie sposób i zakres oddziaływania neuronów na siebie podczas uczenia. Najczęściej stosowane topologie, to topologia liniowa oraz planarne: prostokątna i heksagonalna. Wokół zwycięskiego neuronu definiuje się tzw. topologiczne

sąsiedztwo N_c o określonym promieniu. Promień sąsiedztwa jest zmniejszany podczas uczenia. Przykład sąsiedztwa w formie łańcuch został pokazany na Rys. 6.2.1.

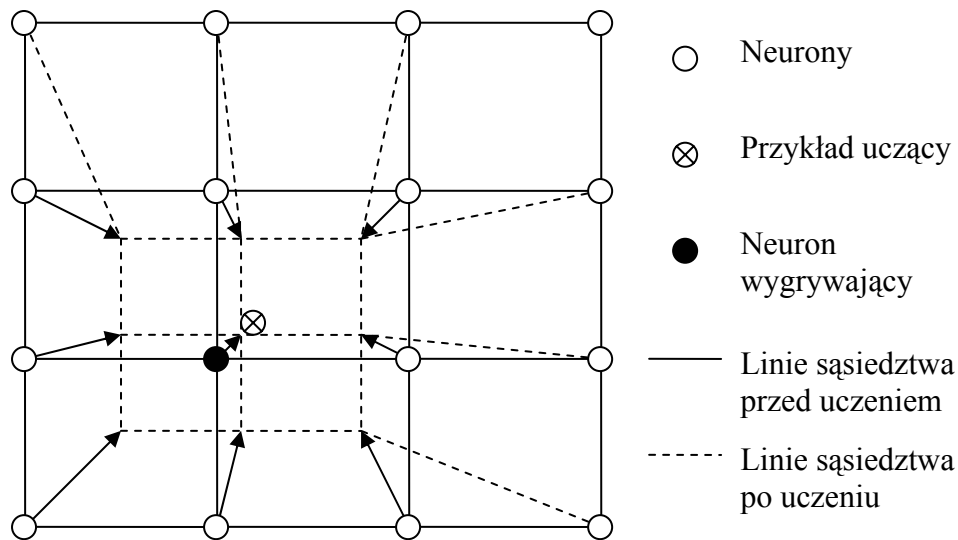


Rys. 6.2.1. Przykład SOM o 2 wejściach i 6 neuronach w topologii liniowej; zaznaczono sąsiedztwo m_3 o trzech różnych promieniach (w trzech różnych chwilach czasowych)

Podczas uczenia na wejście sieci podawane są cyklicznie kolejne przykłady uczące. W chwili t , neuron wygrywający c oraz neurony należące do jego sąsiedztwa podlegają adaptacji, zmieniając swoje wektory wag w taki sposób, aby były one bliższe prezentowanemu wektorowi uczącemu. Adaptacja wag następuje zgodnie ze wzorem

$$\begin{aligned} \mathbf{m}_i(t+1) &= \mathbf{m}_i(t) + \alpha_i(t)[\mathbf{x}(t) - \mathbf{m}_i(t)] && \text{dla każdego } i \in N_c(t), \\ \mathbf{m}_i(t+1) &= \mathbf{m}_i(t) && \text{dla każdego } i \notin N_c(t), \end{aligned} \quad (6.3)$$

gdzie $\alpha_i(t)$ jest współczynnikiem uczenia dla neuronu i w chwili t . Wartość współczynnika uczenia zależy od odległości neuronu od zwycięzcy (w sensie przyjętej metryki i wykorzystywanej topologii) i maleje w czasie uczenia. Proces uczenia dla sieci o dwóch wejściach i topologii prostokątnej przedstawiono na Rys. 6.2.2.



Rys. 6.2.2. Przebieg uczenia dla sieci o 2 wejściach, 16 neuronach i topologii planarnej prostokątnej

Proces nauki prowadzi do organizacji neuronów (adaptacji wag) w ten sposób, aby odwzorować strukturę wzorców prezentowanych podczas nauki. Przestrzeń danych dzielona

jest na obszary przyciągania poszczególnych neuronów. Według niektórych autorów (patrz [1]), do spójnego podziału przestrzeni danych wymagana jest normalizacja wektorów wejściowych lub wektorów wag. Przy założeniu, że wektory uczące są znormalizowane, podczas uczenia następuje automatyczna normalizacja wektorów wag, co wynika z ich przyciągania przez wektory wejściowe.

Istnieją różne metody normalizacji. Najprostszą z nich jest podział każdej składowej wektora wejściowego przez długość tego wektora. Inna metoda, szczegółowo opisana w [1] i [3] zakłada wprowadzenie dodatkowej składowej do wektora o wartości takiej, aby długość wektora wynikowego była równa jeden. Metoda ta ma tę zaletę, że nie odrzuca informacji o wielkościach bezwzględnych zmiennych, ale ma też poważną wadę, jaką jest konieczność przeskalowania wektorów wejściowych, co nie zawsze jest proste.

Należy dodać, że istnieje wiele modyfikacji podstawowego algorytmu Kohonena, wiele podejść do topologii i sąsiedztwa. Jednym z ciekawszych jest użycie sąsiedztwa definiowanego dynamicznie (patrz [4]). Przy takim podejściu topologię sieci określa minimalne drzewo rozpinające, którego wierzchołki reprezentowane są przez wektory wag neuronów. Sieci o dynamicznym sąsiedztwie uważane są za sieci bardziej elastyczne przy adaptacji do zmian w rozkładzie prawdopodobieństwa wartości wejściowych.

6.3. Klasyfikator LVQ

Odmianą sieci samoorganizujących, używaną do klasyfikacji, jest sieć LVQ. W przypadku tego typu sieci zakłada się, że każdy neuron związany jest z jedną z klas. Po podaniu wzorca x na wejście sieci LVQ zostaje on zaklasyfikowany do klasy, z którą związany jest neuron wygrywający, tzn. najbliższy w sensie stosowanej metryki. Podczas uczenia każdy wektor wejściowy musi także mieć przypisaną klasę, której odpowiada. Informacja ta używana jest do odpowiedniej modyfikacji wag neuronów sieci. W podstawowych wersjach algorytmu LVQ w każdym kroku uczenia uaktualnia się wagi jednego lub najwyżej dwóch najbliższych neuronów. Zazwyczaj nie używa się przedstawionej w poprzednim punkcie idei sąsiedztwa, choć prace nad takimi algorytmami też były prowadzone (patrz [5]).

6.4. Metody uczenia

6.4.1. LVQ1

LVQ1 jest podstawową, a równocześnie najbardziej znaną i najczęściej spotykaną w literaturze (np. w [1]) wersją algorytmu uczącego sieć LVQ zaproponowaną przez Kohonena. Algorytm ten zakłada, że wektor uczący x podawany jest na wejście sieci w celu wyłonienia zwycięzcy, czyli neuronu c , którego wagi są najbliższe wektorowi x . Wagi zwycięzcy są następnie uaktualniane w sposób zależny od tego, czy wektor uczący i neuron związane są z tą samą klasą. W przypadku, gdy klasy wektora x i neuronu są takie same, następuje przesunięcie wag neuronu w kierunku wektora wejściowego. W przeciwnym przypadku, wektor wag odsuwany jest od wektora wejściowego. Pozostałe neurony sieci nie są modyfikowane. Algorytm ten można przedstawić następująco

$$\begin{aligned} \mathbf{m}_c(t+1) &= \mathbf{m}_c(t) + \alpha(t)[\mathbf{x}(t) - \mathbf{m}_c(t)] && \text{jeśli } \mathbf{x} \text{ i } \mathbf{m}_c \text{ należą do tej samej klasy,} \\ \mathbf{m}_c(t+1) &= \mathbf{m}_c(t) - \alpha(t)[\mathbf{x}(t) - \mathbf{m}_c(t)] && \text{jeśli } \mathbf{x} \text{ i } \mathbf{m}_c \text{ należą do różnych klas,} \\ \mathbf{m}_i(t+1) &= \mathbf{m}_i(t) && \text{dla } i \neq c, \end{aligned} \quad (6.4)$$

gdzie $0 < \alpha(t) < 1$ oraz $\alpha(t)$ jest malejącym w czasie współczynnikiem uczenia. Kohonen w pracy [4] sugeruje raczej niewielką początkową wartość współczynnika α , np. $\alpha_0 = 0,1$.

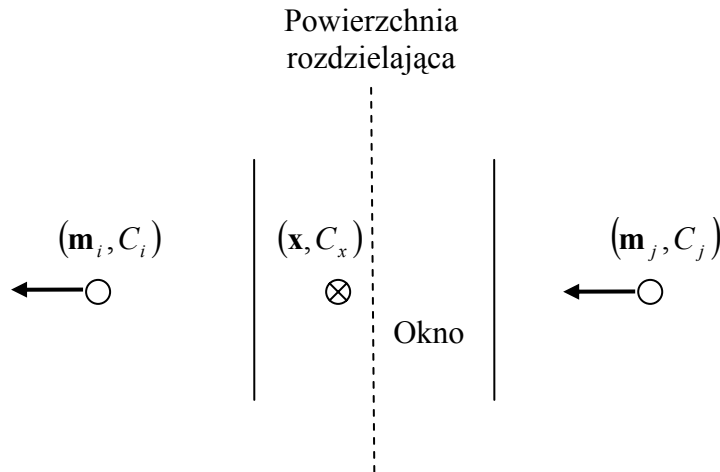
Podobne wartości sugerują inni autorzy, patrz [1]. Współczynnik uczenia może być zmniejszany liniowo lub wykładniczo do zera.

6.4.2. LVQ2

Drugą metodą uczenia sieci LVQ zaproponowaną przez Kohonena jest metoda nazywana LVQ2. W tym przypadku zakłada się, że aktualizacja wag następuje równocześnie dla dwóch najbliższych neuronów \mathbf{m}_i oraz \mathbf{m}_j , przy czym muszą być spełnione odpowiednie warunki. Najbliższy neuron, oznaczony przez \mathbf{m}_i , musi należeć do innej klasy niż wejściowy przykład uczący. Drugi najbliższy neuron \mathbf{m}_j musi należeć do tej samej klasy, co \mathbf{x} . Ponadto \mathbf{x} musi wpadać w symetryczne okno, które jest zdefiniowane wokół powierzchni rozdzielającej dzielącej obszary decyzyjne odpowiadające neuronom \mathbf{m}_i oraz \mathbf{m}_j . Biorąc pod uwagę wszystkie warunki, \mathbf{x} musi wpadać w okno po niewłaściwej stronie powierzchni rozdzielającej. Jeżeli przez d_i i d_j oznaczymy odległości między \mathbf{x} i odpowiednio \mathbf{m}_i oraz \mathbf{m}_j , to przyjmuje się, że \mathbf{x} wpada do okna o względnej szerokości w , jeżeli spełniona jest zależność

$$\min\left(\frac{d_i}{d_j}, \frac{d_j}{d_i}\right) > s, \text{ gdzie } s = \frac{1-w}{1+w}. \quad (6.5)$$

W pracy [4] zaleca się względną szerokość okna w równą 0,2 lub 0,3.



Rys. 6.4.1. Ilustracja procesu nauki metodą LVQ2; strzałkami oznaczono kierunek zmian wag w przypadku, gdy klasa przykładu \mathbf{x} odpowiada klasie C_j i jest niezgodna z C_i

Zgodnie z [3] algorytm uaktualniania wag można przedstawić następująco

$$\begin{aligned} \mathbf{m}_i(t+1) &= \mathbf{m}_i(t) - \alpha(t)[\mathbf{x}(t) - \mathbf{m}_i(t)], \\ \mathbf{m}_j(t+1) &= \mathbf{m}_j(t) + \alpha(t)[\mathbf{x}(t) - \mathbf{m}_j(t)], \end{aligned} \quad (6.6)$$

jeśli C_i jest najbliższą klasą, ale \mathbf{x} należy do klasy $C_j \neq C_i$, gdzie C_j jest drugą najbliższą klasą, ponadto \mathbf{x} wpada w okno. Przypadek ten został przedstawiony na Rys. 6.4.1. W każdym innym przypadku zachodzi

$$\mathbf{m}_k(t+1) = \mathbf{m}_k(t). \quad (6.7)$$

Poprawki na wagi dokonywane są w ten sposób, że powierzchnia rozdzielająca obszarów decyzyjnych przesuwa się w tak, aby zminimalizować prawdopodobieństwo błędnej klasyfikacji.

W przypadku tego algorytmu ważne jest, aby początkowe wagi neuronów miały wartości dobrane w ten sposób, że neurony leżą wewnątrz prawidłowych obszarów decyzyjnych. Można to osiągnąć przez wstępną analizę zbioru uczącego i odpowiednią inicjalizację wag, tak jak pokazano to w 6.5.1 lub poprzez wykonanie pierwszej fazy nauki algorytmem LVQ1, który pod tym względem jest mniej restrykcyjny. Algorytm LVQ2 służy jedynie do dopasowania wstępnie ustalonych powierzchni rozdzielających tak, aby zminimalizować błąd.

6.4.3. LVQ3

Trzecia wersja algorytmu uczenia sieci LVQ, oznaczona przez LVQ3, wprowadza dodatkowe modyfikacje do wersji poprzednich. Poprzedni algorytm przesuwiał granicę decyzyjną nie biorąc pod uwagę, co może się stać z położeniem \mathbf{m}_i przy długim działaniu. LVQ3 wprowadza dodatkowe korekty, które zapewniają, że \mathbf{m}_i w kolejnych iteracjach nadal dobrze aproksymuje rozkład klas. Przy założeniach takich jak 6.4.2, tzn. \mathbf{m}_i oraz \mathbf{m}_j są dwoma najbliższymi neuronami względem wektora uczącego \mathbf{x} , przy czym \mathbf{x} i \mathbf{m}_j należą do tej samej klasy, a \mathbf{x} i \mathbf{m}_i należą do różnych klas, ponadto \mathbf{x} wpada w okno o względnej szerokości w , algorytm LVQ3 można przedstawić następująco:

$$\begin{aligned}\mathbf{m}_i(t+1) &= \mathbf{m}_i(t) - \alpha(t)[\mathbf{x}(t) - \mathbf{m}_i(t)] \\ \mathbf{m}_j(t+1) &= \mathbf{m}_j(t) + \alpha(t)[\mathbf{x}(t) - \mathbf{m}_j(t)]\end{aligned}\quad (6.8)$$

oraz

$$\mathbf{m}_k(t+1) = \mathbf{m}_k(t) + \varepsilon\alpha(t)[\mathbf{x}(t) - \mathbf{m}_k(t)]$$

dla $k \in \{i, j\}$ w przypadku, gdy \mathbf{x} , \mathbf{m}_i oraz \mathbf{m}_j należą do tej samej klasy. Zgodnie z sugestiami przedstawionymi w pracy [5], współczynnik ε powinien przyjmować wartość od 0,1 do 0,5, w zależności od szerokości okna.

Podobnie jak w przypadku algorytmu LVQ2, w tym wypadku także ważne jest początkowe położenie neuronów. Zaleca się użycie algorytmu LVQ3 tylko do końcowego dopasowania granic obszarów decyzyjnych.

6.4.4. Modyfikacje

W pracy [5] Kohonen zaproponował podstawową modyfikację algorytmu uczenia sieci LVQ poprzez optymalne dobieranie współczynnika uczenia. Modyfikację tę można zastosować w algorytmach LVQ1 oraz LVQ3. Zmodyfikowane algorytmy nazywa się odpowiednio OLVQ1 oraz OLVQ3 (z ang. Optimized-Learning-Rate LVQ).

W modyfikacji tej zakłada się, że z każdym neuronem i o wektorze wag \mathbf{m}_i związany jest indywidualny współczynnik uczenia $\alpha_i(t)$. W takim przypadku podstawowa zasada nauki LVQ dla neuronu wygrywającego c przyjmuje postać:

$$\begin{aligned}\mathbf{m}_c(t+1) &= \mathbf{m}_c(t) + \alpha_c(t)[\mathbf{x}(t) - \mathbf{m}_c(t)] && \text{jeśli } \mathbf{x} \text{ i } \mathbf{m}_c \text{ należą do tej samej klasy,} \\ \mathbf{m}_c(t+1) &= \mathbf{m}_c(t) - \alpha_c(t)[\mathbf{x}(t) - \mathbf{m}_c(t)] && \text{jeśli } \mathbf{x} \text{ i } \mathbf{m}_c \text{ należą do różnych klas,} \\ \mathbf{m}_i(t+1) &= \mathbf{m}_i(t) && \text{dla } i \neq c.\end{aligned}\quad (6.9)$$

Celem tej metody jest dobieranie współczynnika uczenia w taki sposób, aby przyspieszyć zbieżność. Powyższe równania można wyrazić w następujący sposób:

$$\mathbf{m}_c(t+1) = [1 - s(t)\alpha_c(t)]\mathbf{m}_c(t) + s(t)\alpha_c(t)\mathbf{x}(t), \quad (6.10)$$

gdzie $s(t)=1$ w przypadku prawidłowej klasyfikacji oraz $s(t)=-1$ w przypadku błędnej klasyfikacji. Można zauważyć, że wektor $\mathbf{m}_c(t+1)$ składa się z dwóch części:

przeskalowanego wektora wejściowego oraz przeskalowanego poprzedniego wektora wag. Analogicznie, poprzedni wektor wag $\mathbf{m}_c(t)$ zawiera składniki związane z wcześniejszymi wektorem wag oraz przykładem uczącym. Ogólnie można stwierdzić, że $\mathbf{m}_c(t+1)$ zawiera ślad wszystkich wektorów uczących podawanych na wejście sieci. Ideą tej metody jest zapewnienie, aby każdy z przykładów uczących \mathbf{x} wносił taki sam wkład w proces nauki. Jeśli zauważymy, że wektor $\mathbf{m}_c(t+1)$ zawiera wektor $\mathbf{x}(t)$ przeskalowany o $\alpha_c(t)$, oraz wektor $\mathbf{x}(t-1)$ przeskalowany o wartość $[1 - s(t)\alpha_c(t)]\alpha_c(t-1)$, możemy żądać, aby te dwie wartości były sobie równe, tj.

$$\alpha_c(t) = [1 - s(t)\alpha_c(t)]\alpha_c(t-1), \quad (6.11)$$

stąd optymalny współczynnik uczenie, tzn. taki, który zapewnia równomierny wkład każdego przykładu uczącego w proces nauki, można określić zależnością rekurencyjną:

$$\alpha_c(t) = \frac{\alpha_c(t-1)}{1 + s(t)\alpha_c(t-1)}. \quad (6.12)$$

W pracy [5] zaleca się przyjęcie początkowego współczynnika uczenia o wartości równej ok. 0,3 lub 0,5. Zwraca się także uwagę na to, że opisany powyżej sposób obliczania współczynnika uczenia może powodować jego wzrost. W związku z tym należy zabezpieczyć się przed wzrostem powyżej wartości 1, co mogłoby prowadzić do braku zbieżności.

6.5. Uwagi dotyczące uczenia

6.5.1. Inicjalizacja wag początkowych

Zadaniem algorytmu LVQ nie jest aproksymacja funkcji gęstości klas dla przykładów uczących, lecz ustalenie granic obszarów decyzyjnych zgodnych z regułą najbliższy sąsiad. Wyniki, jakie można uzyskać, zależą w dużej mierze od wartości początkowych wektorów wag neuronów, przy czym w przypadku algorytmów LVQ2 i LVQ3 wartości te mają kluczowe znaczenie. Mniejszy wpływ na wyniki ma liczba neuronów przypisana do każdej klasy. Zaleca się rozpoczęcie uczenia dla takiej samej liczby neuronów w każdej klasie.

Najprostszą metodą inicjalizacji wag jest metoda losowa. Jednak z powodów wspomnianych w punktach 6.4.2 oraz 6.4.3, nie jest ona odpowiednia przy stosowaniu algorytmów LVQ2 lub LVQ3. Inną metodą inicjalizacji wag, przedstawioną w pracy [5], jest inicjalizacja na podstawie próbek ze zbioru uczącego. Polega ona na tym, że ze zbioru uczącego wybiera się odpowiednią ilość przykładów dla każdej z klas i na ich podstawie inicjalizuje się wagi neuronów. Ponieważ wektory wag powinny leżeć wewnątrz obszarów decyzyjnych odpowiadających danym klasom, ważne jest, aby do inicjalizacji wybierać przykłady, które są poprawnie klasyfikowane (np. metodą k-NN) względem pozostałej części zbioru uczącego. Po takim zainicjalizowaniu wag, granice obszarów decyzyjnych są wstępnie ustalone, ale wymagają dopasowania do pozostałych danych uczących.

6.5.2. Przebieg uczenia

Przebieg uczenia może wyglądać w różny sposób. W pracy [5] zaleca się rozpoczęcie nauki od metody OLVQ1, która charakteryzuje się szybką zbieżnością, a następnie po określonej liczbie iteracji, przejście do metody LVQ1, LVQ2 lub LVQ3 z małym współczynnikiem uczenia. Ogólnie zaleca się wstępną inicjalizację wag za pomocą metody opisanej w punkcie 6.5.1, a w przypadku korzystania tylko z algorytmów LVQ2 lub LVQ3, jest to wręcz konieczne. Liczba niezbędnych iteracji dla LVQ1-3 wynosi 50-200 razy liczba neuronów, a w przypadku algorytmu OLVQ1 30-50 razy liczba neuronów.

Aktualizacji wag można następować bezpośrednio po podaniu każdego z przykładów uczących lub jak sugeruje się w pracy [3], obliczając średnią poprawkę dla każdego neuronu po całym zbiorze uczącym.

6.5.3. Problem nieużywanych neuronów

Podczas procesu uczenia, szczególnie w przypadku, gdy wagi początkowe były wybrane losowo, może zaistnieć sytuacja, w której niektóre neurony nigdy nie wygrywają, a przez to nigdy nie są poddawane procesowi adaptacji wag. Istnieje wiele sposobów poradzenia sobie z tym problemem. Jednym z prostszych, a zarazem bardzo skutecznym jest sposób prezentowany w [3]. Polega on na tym, że część ciężaru przenosi się z neuronów, które próbują prezentować zbyt szeroką gamę wzorców na nieużywane neurony. Można tego dokonać w trzech krokach:

- Przechodząc przez cały zbiór uczący wyznacz dla każdego przypadku zwycięski neuron (o największej aktywacji, najmniejszej odległości) równocześnie śledź minimalną wygrywającą aktywację (tj. największą odległość). W wyniku otrzymujemy przypadek najgorzej reprezentowany w sieci.
- Z neuronów, które ani razu nie wygrały w poprzednim kroku (więc nie są używane) wybierz taki, którego aktywacja jest największa dla prezentowanego wzorca znalezionego w kroku pierwszym (tj. taki, którego wagi są najbliższe prezentowanemu wektorowi uczącemu).
- Ustal wektor wag wybranego neuronu w taki sposób, aby był równy przykładowi uczącemu wyłonionemu w kroku pierwszym.

Po wykonaniu powyższych kroków, najgorzej reprezentowanemu przykładowi w zbiorze uczącym przypisywany jest neuron, który wcześniej nie był używany. Metoda ta nie wprowadza zaburzeń, jeśli wszystkie neurony są używane. Po dokonaniu modyfikacji należy ponownie wystartować algorytm adaptacji wag.


Przedstawiona powyżej metoda dotyczy uczenia sieci SOM, ale po pewnych modyfikacjach można ją zastosować także dla sieci LVQ. Zmodyfikowany algorytm przyjmuje postać:



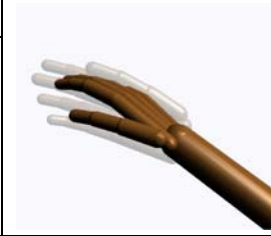

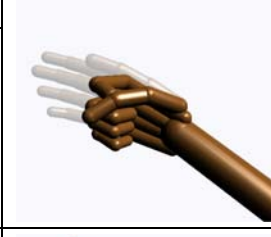

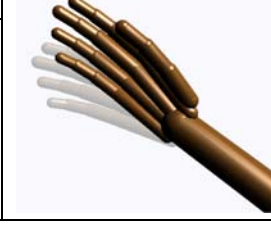
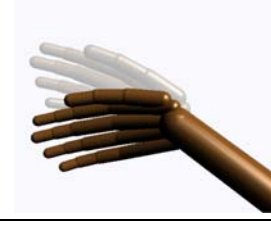
- Przechodząc przez cały zbiór uczący wyznacz przykład uczący, który nie został poprawnie zaklasyfikowany i jest najbardziej oddalony od najbliższego reprezentanta swojej klasy.
- Z neuronów, które w poprzednim kroku nie wygrały ani razu lub wygrywały prowadząc do błędnej klasyfikacji, wybierz taki, którego aktywacja jest największa dla prezentowanego wzorca znalezionego w kroku pierwszym.
- Ustal wektor wag wybranego neuronu tak, aby był równy przykładowi uczącemu wyłonionemu w kroku pierwszym i jeśli to konieczne zmień klasę przypisaną neuronowi.

7. Wyniki eksperymentów

7.1. Wyróżnione klasy

Podczas realizacji pracy założono, że rozpatrywanych będzie dziewięć różnych stanów dłoni, co oznacza, że przewiduje się występowanie dziewięciu klas w systemie rozpoznawania. Klasa zerowa odpowiada stanowi neutralnemu. Stany dłoni, wraz z numerami i etykietami klas oraz opisem zebrane są w poniższej tabeli.

0 – Rest	
Podstawowa, swobodna, neutralna pozycja dłoni	

1 - Flexion		2 - Extension	
Zgięcie dłoni w nadgarstku, palce w pozycji neutralnej		Wygięcie dłoni w nadgarstku, palce w pozycji neutralnej	
3 – Pronation		4 – Supination	
Przywodzenie dłoni, wnętrze dłoni skierowane w dół, palce w pozycji neutralnej		Odwodzenie dłoni, wnętrze dłoni skierowane w górę, palce w pozycji neutralnej	
5 – Finger flexion		6 – Finger extension	
Zamknięcie dłoni w pięść, nadgarstek w pozycji neutralnej		Całkowite otwarcie dłoni, palce wyprostowane, nadgarstek w pozycji neutralnej	
7 – Radial deviation		8 – Ulnar deviation	
Podniesienie dłoni, palce w pozycji neutralnej		Opuszczenie dłoni, palce w pozycji neutralnej	

7.2. Pomiar sygnałów

7.2.1. Ułożenie elektrod

Do pomiaru miopotencjałów używane były typowe elektrody EKG oraz wzmacniacze, których budowa przedstawiona jest w rozdziale 4.2. Pomiary wykonano dla dwóch ułożeń elektrod pomiarowych. W obu przypadkach elektrody kanału pierwszego umieszczone były na grupie mięśni prostujących nadgarstek i palce (przednia powierzchnia przedramienia), a elektrody kanału drugiego na grupie mięśni zginających nadgarstek i palce (tylna powierzchnia przedramienia). Elektroda odniesienia umieszczona była w okolicach nadgarstka (tkanka elektrycznie obojętna).

W przypadku pierwszego sposobu ułożenia, oznaczonego przez **E1**, elektrody umieszczone były wzdłuż mięśni, co przedstawia Rys. 7.2.1. Przy drugim sposobie ułożenia, oznaczonym przez **E2** i przedstawionym na Rys. 7.2.2, elektrody umieszczone były w poprzek mięśni.



Rys. 7.2.1. Pierwszy sposób umieszczenia elektrod – E1



Rys. 7.2.2. Drugi sposób umieszczenia elektrod – E2

7.2.2. Zamiana na postać cyfrową

Zamiany sygnałów na postać cyfrową dokonano za pomocą przetworników analogowo-cyfrowych wchodzących w skład karty dźwiękowej *Sound Blaster Audigy* firmy Creative. Wzmacniacze podłączono do wejścia sygnałowego (line-in) karty. Wszystkie pomiary wykonano przy następujących parametrach próbkowania:

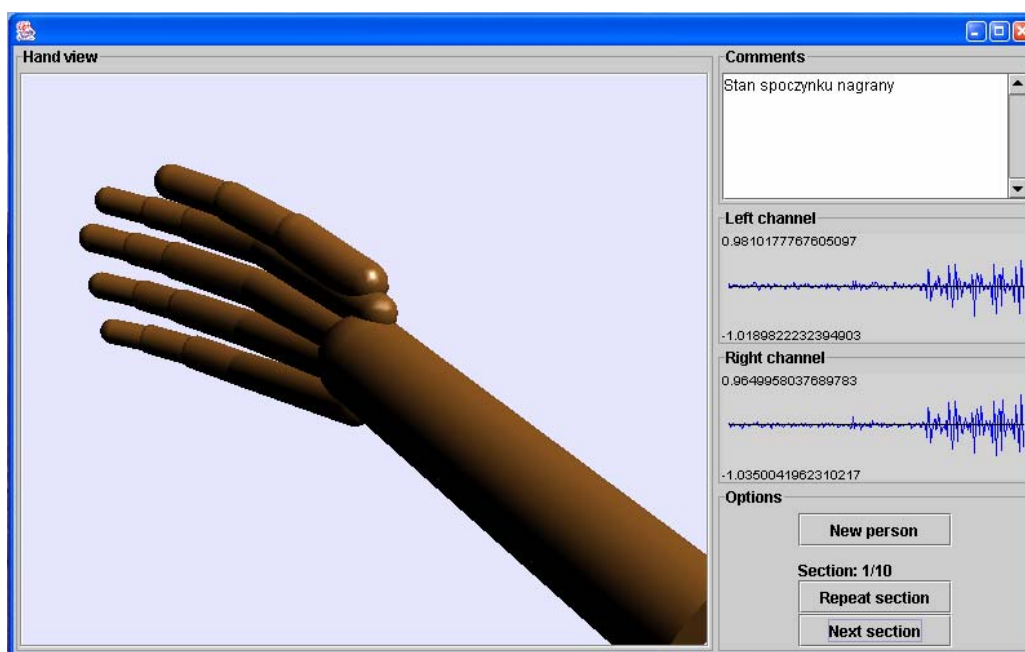
- rozdzielczość 16 bitów,
- rejestracja stereofoniczna,
- częstotliwość próbkowania $f = 8$ kHz.

Wybrano najniższą dostępną częstotliwość próbkowania. Spróbkowane sygnały zapisane zostały w formie plików dźwiękowych WAVE (o rozszerzeniu „wav”).

7.2.3. Sposób tworzenia bazy pomiarów

Na potrzeby pracy stworzono bazę sygnałów zmierzonych dla 4 zdrowych osób, oznaczonych kolejno **O1**, **O2**, **O3** i **O4**. Baza zawiera sygnały z mięśni mierzone dla dwóch ułożeń elektrod przedstawionych w punkcie 7.2.1. Dla każdej osoby i każdego ułożenia elektrod wszystkie pomiary zostały przeprowadzone dwukrotnie w odstępach półgodzinnych, co w wyniku dało cztery zbiory plików na osobę.

Do wykonania pomiarów użyto program *EMGRecorder*, który jest opisany w dodatku B. Widok okna programu przedstawiono na Rys. 7.2.3. Zadaniem programu było przeprowadzenie pomiarów według danego schematu w sposób jednolity i spójny. Każdej z badanych osób prezentowane były kolejne stany dłoni, które osoba ta miała naśladować. Jednocześnie program dokonywał nagrań dzieląc w sposób automatyczny zapis sygnału na zdefiniowany zbiór plików.



Rys. 7.2.3. Widok okna programu EMGRecorder

Zbiór plików dla jednej osoby przy ustalonym sposobie ułożenia elektrod tworzony był według następującego schematu:

- pierwszy plik zawiera 5 sekundowy zapis stanu spoczynku,
- dla każdego innego stanu zarejestrowano po pięć plików, przy czym każdy plik zawiera zapis 6 sekundowy: 1 sekunda to ruch ze stanu spoczynku do konkretnego stanu, kolejne 5 sekund to zapis miopotencjałów w danym stanie, odstęp między kolejnymi nagraniami wynosił 2 sekundy,
- dodatkowo zarejestrowano plik zawierający zapis wszystkich stanów, przy czym plik ma następującą strukturę: przejście ze stanu spoczynku do kolejnego stanu wykonywane było w ciągu 1 sekundy, stan utrzymywał się 5 sekund, przejście do stanu spoczynku – 1 sekunda, odstęp między stanami – 1 sekunda.

Kolejność stanów podczas zapisu odpowiadała numeracji klas zdefiniowanej w punkcie 7.1. Podsumowując, dla każdej osoby zarejestrowano cztery zbiory plików, każdy zbiór zawiera po pięć serii pomiarów (podzielonych na pliki odpowiadające poszczególnym klasom).

7.3. Zbiór uczący i testujący

Otrzymana baza sygnałów stanowi podstawę wszystkich następnych badań. Z każdej serii pomiarów, wydobyto zbiór uczący i testujący. Zbiory te tworzone były dla następujących parametrów:

- częstotliwość próbkowania $f = 1600$ Hz,
- szerokość okna (patrz punkt 4.3.1) $N = 256$,
- funkcja okienkująca: Hamminga
- współczynnik korekcji (patrz punkt 4.4) $\gamma = 0,6$,
- liczba punktów uśredniania (patrz punkt 4.4) $IP = 10$,
- liczba cech na kanał $E = 8$,
- liczba kanałów $L = 2$,
- liczba cech $d = 16$,
- liczba klas $c = 9$.

Zbiór uczący utworzony z pojedynczej serii pomiarów zawiera 10 przykładów dla każdej klasy, co w sumie daje 90 przykładów uczących. Zbiór testujący zawiera po 20 przykładów dla każdej klasy (łącznie 180).

7.4. Sieci MLP

Poniżej przedstawiono wyniki przeprowadzonych badań dotyczących doboru parametrów uczenia sieci. W pracy przyjęto następujące oznaczenia:

- EU – średni procentowy błąd rozpoznawania zbioru uczącego,
- DU – odchylenie standardowe błędu rozpoznawania zbioru uczącego,
- ET – średni procentowy błąd rozpoznawania zbioru testującego,
- DT – odchylenie standardowe błędu rozpoznawania zbioru testującego.

7.4.1. Dobór struktury sieci

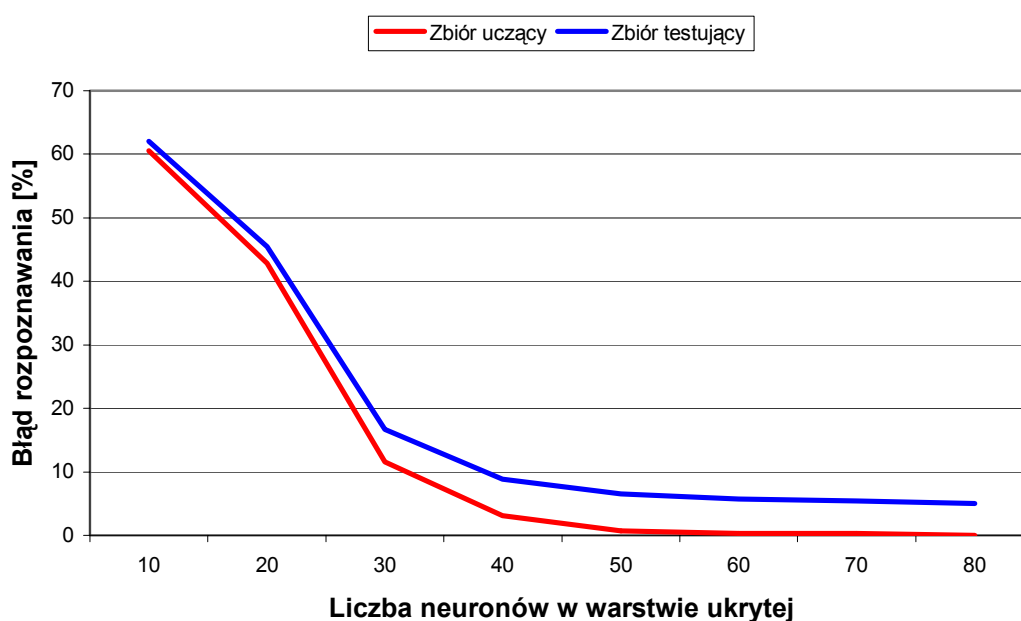
Stosując podstawowy algorytm uczenia sieci wielowarstwowej ze stałym współczynnikiem uczenia (przedstawiony w 5.7.1) oraz momentum (patrz 5.6) wyznaczono liczbę neuronów warstwy ukrytej, dla której błąd klasyfikacji osiąga zadowalająco małą wartość. Eksperymenty przeprowadzono dla następujących parametrów:

- zbiór uczący: 90 elementów (po 10 na każdą klasę), utworzony na podstawie jednej serii pomiarów dla jednej osoby,
- zbiór testujący: 180 elementów (po 20 na każdą klasę), utworzony na podstawie jednej serii pomiarów dla jednej osoby,
- współczynnik uczenia $\eta = 0,001$,
- momentum $\alpha = 0,7$,
- ograniczenie liczby iteracji: 1000.

Tabela 7.4.1 zawiera wartości średnie i odchylenia standardowe z dziesięciu pomiarów błędu rozpoznawania dla różnych ilości neuronów w warstwie ukrytej.

Liczba neuronów	Zbiór uczący		Zbiór testujący	
	EU [%]	DU	ET [%]	DT
10	60,56	14,05	62,06	12,89
20	42,89	17,04	45,50	15,26
30	11,56	10,49	16,72	8,64
40	3,11	3,58	8,89	3,89
50	0,67	0,78	6,50	2,13
60	0,33	0,75	5,78	0,88
70	0,33	1,05	5,44	1,17
80	0,00	0,00	5,00	0,39

Tabela 7.4.1. Średni błąd rozpoznawania oraz odchylenie standardowe błędów w zależności od liczby neuronów



Rys. 7.4.1. Średni błąd rozpoznawania w zależności od liczby neuronów w warstwie ukrytej

Na podstawie wyników można stwierdzić, że dla badanego problemu minimalna liczba neuronów w warstwie ukrytej wynosi 50. Dla takiej ilości błąd rozpoznawania na zbiorze uczącym spadł poniżej 1%, a na zbiorze testującym do wartości ok. 6,5%. Należy zauważyć, że większa liczba neuronów w warstwie ukrytej powoduje zwiększenie czasu potrzebnego na naukę.

7.4.2. Dobór współczynnika uczenia i momentum dla MLP1

Zbadano szybkość uczenia przy zastosowaniu algorytmu MLP1 (patrz punkt 5.7.1) dla różnych wartości współczynnika uczenia η oraz momentum α . Pomiary przeprowadzono w następujących warunkach:

- zbiór uczący: 90 elementów (po 10 na każdą klasę), utworzony na podstawie jednej serii pomiarów dla jednej osoby,
- zbiór testujący: 180 elementów (po 20 na każdą klasę), utworzony na podstawie jednej serii pomiarów dla jednej osoby,
- liczba neuronów w warstwie ukrytej: 40,

- ograniczenie liczby iteracji: 500.

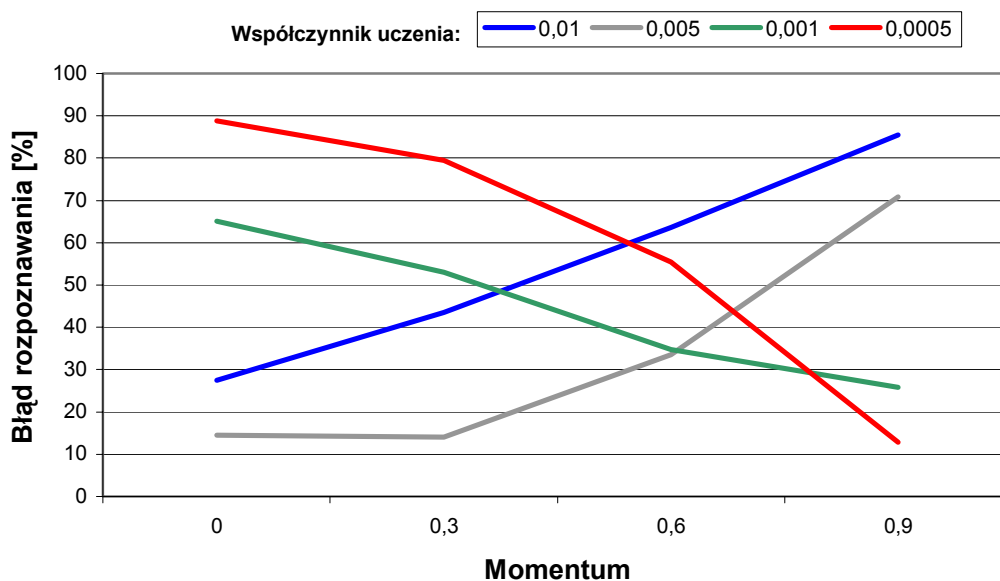
Tabela 7.4.2 zawiera wartości średnie oraz odchylenia standardowe z dziesięciu pomiarów błędu rozpoznawania zbioru uczącego dla różnych wartości współczynnika uczenia oraz momentum. Tabela 7.4.3 zawiera analogiczne wartości dla zbioru testującego. Wyróżniono wartości najmniejsze.

Współczynnik uczenia η	Momentum α							
	0		0,3		0,6		0,9	
	EU [%]	DU	EU [%]	DU	EU [%]	DU	EU [%]	DU
0,01	27,56	10,50	43,56	10,83	63,56	12,44	85,56	5,37
0,005	14,56	10,75	14,00	6,91	33,56	12,71	70,89	9,35
0,001	65,11	13,10	53,00	9,47	34,78	12,21	25,78	17,90
0,0005	88,89	0,00	79,44	7,63	55,50	11,70	12,78	7,05

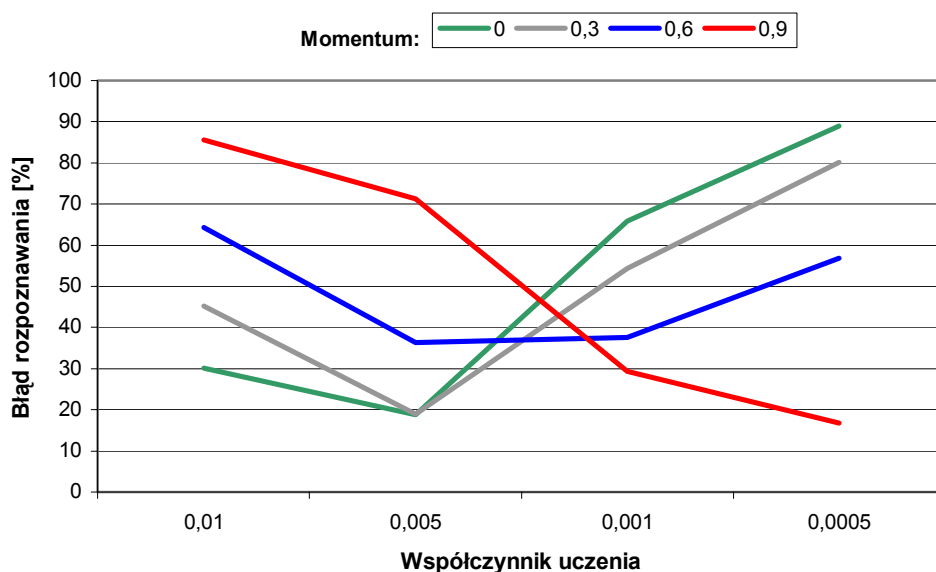
Tabela 7.4.2. Średni błąd rozpoznawania i odchylenie standardowe na zbiorze uczącym w zależności od wartości współczynnika uczenia i momentum

Współczynnik uczenia η	Momentum α							
	0		0,3		0,6		0,9	
	EU [%]	DU	EU [%]	DU	EU [%]	DU	EU [%]	DU
0,01	30,06	9,58	45,17	9,58	64,22	11,41	85,61	5,28
0,005	18,78	9,03	18,94	5,73	36,28	11,29	71,22	9,04
0,001	65,89	12,89	54,39	9,50	37,56	10,35	29,33	17,25
0,0005	89,00	0,23	80,17	7,14	56,89	11,07	16,83	5,98

Tabela 7.4.3. Średni błąd rozpoznawania i odchylenie standardowe na zbiorze testującym w zależności od wartości współczynnika uczenia i momentum

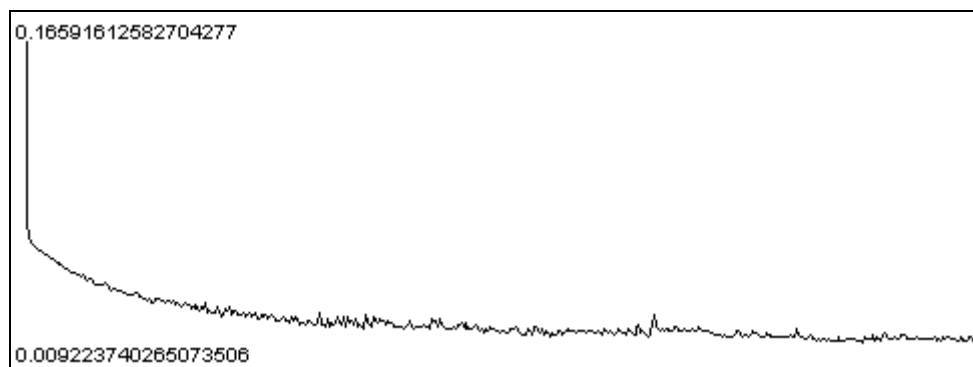


Rys. 7.4.2. Średni błąd rozpoznawania na zbiorze uczącym w zależności od wartości momentum dla różnych wartości współczynnika uczenia



Rys. 7.4.3. Średni błąd rozpoznawania na zbiorze testującym w zależności od współczynnika uczenia dla różnych wartości momentu

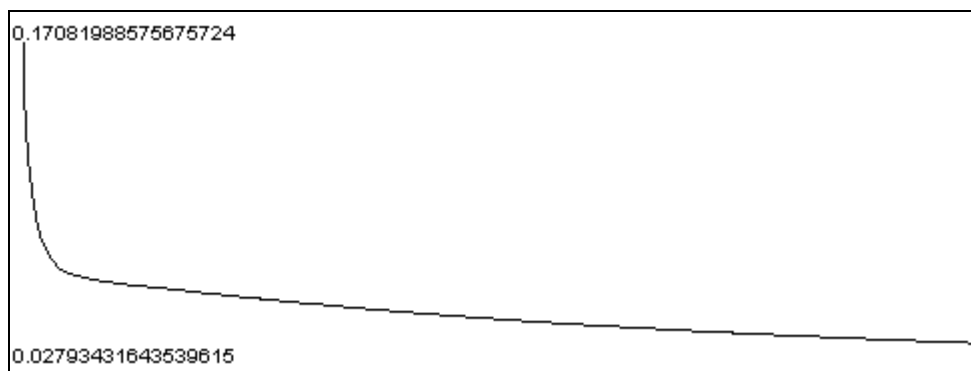
Otrzymane wyniki potwierdzają, że przy zbyt dużej wartości współczynnika uczenia (0,01) trudno jest osiągnąć dobre rezultaty, gdyż następuje przeskakiwanie nad minimum – proces nauki nie jest zbieżny. Na Rys. 7.4.4 można zaobserwować wykres błędu średniokwadratowego sieci podczas nauki ze zbyt dużym współczynnikiem uczenia. Błąd nie maleje jednostajnie, linia wykresu jest poszarpana, co jest związane z przeskakiwaniem minimum.



Rys. 7.4.4. Wykres błędu średniokwadratowego sieci dla 500 kroków nauki przy współczynniku uczenia równym 0,01 i momencie równym 0

Dla współczynnika uczenia równego 0,005, błąd rozpoznawania jest mniejszy (ok. 14% dla zbioru uczącego) i jest to najlepsza osiągnięta wartość przy założeniu, że nie wykorzystuje się momentu.

Zmniejszenie współczynnika uczenia (do 0,001 i 0,0005) spowodowało ponowne zwiększenie błędu rozpoznawania, co związane jest z powolnością procesu minimalizacji błędu. Na Rys. 7.4.5 przedstawiono przykładowy wykres błędu średniokwadratowego sieci podczas nauki z współczynnikiem uczenia równym 0,001. Wykres ma gładką postać, co oznacza, że minimalizacja przeprowadzana jest prawidłowo, bez przeskoków nad minimum, lecz ograniczenie iteracji nie pozwoliło osiągnąć minimum lokalnego.



Rys. 7.4.5. Wykres błędu średniokwadratowego sieci dla 500 kroków nauki przy współczynniku uczenia równym 0,001 i momentum 0

Wprowadzenie momentum powoduje, że przy treningu z dużym współczynnikiem uczenia średni błąd rozpoznawania jest większy, jednak dla małych wartości współczynnika uczenia udaje się znacząco zwiększyć szybkość uczenia. Po 500 iteracjach nauki przy parametrach $\eta = 0,0005$ i $\alpha = 0,9$ sieć była w stanie rozpoznawać zbiór uczący z błędem poniżej 13%, a zbiór testujący z błędem poniżej 17%.

7.4.3. Dobór parametrów dla metody MLP2

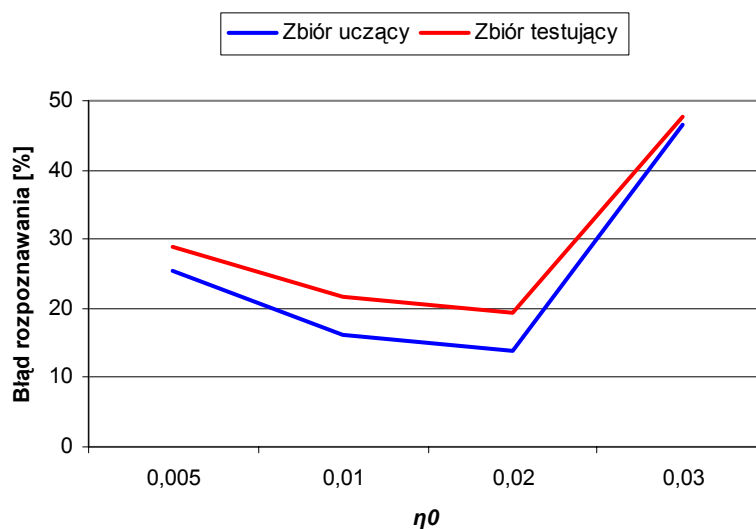
Metoda MLP2 (patrz punkt 5.7.2) wymaga dobrania trzech parametrów: η_0 , η_{95} oraz η_{100} , które są współczynnikami uczenia dla odpowiednio: zerowej, 95-cio procentowej i stuprocentowej skuteczności rozpoznawania. Parametry te dobrano przy następujących warunkach:

- zbiór uczący: 90 elementów (po 10 na każdą klasę), utworzony na podstawie jednej serii pomiarów dla jednej osoby,
- zbiór testujący: 180 elementów (po 20 na każdą klasę), utworzony na podstawie jednej serii pomiarów dla jednej osoby,
- liczba neuronów w warstwie ukrytej: 40,
- ograniczenie liczby iteracji: 500.

Tabela 7.4.4 przedstawia średni błąd rozpoznawania i odchylenie standardowe z dziesięciu pomiarów dla różnych wartości η_0 oraz $\eta_{100} = 0,0001$ i $\eta_{95} = 0,001$. Wyróżniono wartości najmniejsze. Na Rys. 7.4.6 przedstawiono graficznie zależność błędu od wartości η_0 .

η_0	Zbiór uczący		Zbiór testujący	
	EU [%]	DU	ET [%]	DT
0,005	25,44	9,48	28,78	8,42
0,01	16,22	18,07	21,56	16,05
0,02	14,00	9,42	19,33	7,86
0,03	46,67	16,59	47,56	15,86

Tabela 7.4.4. Wartości średnie błędu rozpoznawania i odchylenia standardowe w zależności od η_0 przy $\eta_{100} = 0,0001$ oraz $\eta_{95} = 0,001$

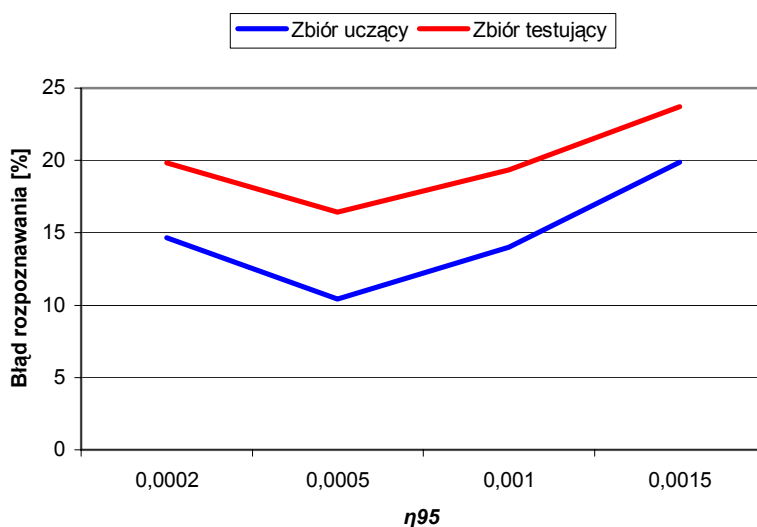


Rys. 7.4.6. Wartości średnie błędu rozpoznawania w zależności od η_0 przy $\eta_{100} = 0,0001$ oraz $\eta_{95} = 0,001$

Błąd rozpoznawania osiąga najmniejszą wartość dla $\eta_0 = 0,02$. Tabela 7.4.5 przedstawia średni błąd rozpoznawania i odchylenie standardowe dla różnych wartości η_{95} , przy ustalonym $\eta_{100} = 0,0001$ oraz $\eta_0 = 0,02$. Wartości minimalne zostały wyróżnione.

η_{95}	Zbiór uczący		Zbiór testujący	
	EU [%]	DU	ET [%]	DT
0,0002	14,67	10,65	19,83	9,42
0,0005	10,44	6,87	16,44	5,25
0,001	14,00	9,42	19,33	7,86
0,0015	19,89	12,15	23,72	10,26

Tabela 7.4.5. Wartości średnie błędu rozpoznawania i odchylenia standardowe w zależności od η_{95} przy $\eta_{100} = 0,0001$ oraz $\eta_0 = 0,02$



Rys. 7.4.7. Wartości średnie błędu rozpoznawania w zależności od η_{95} przy $\eta_{100} = 0,0001$ oraz $\eta_0 = 0,02$

Podsumowując, najmniejszy błąd rozpoznawania osiągnięto dla $\eta_0 = 0,02$, $\eta_{95} = 0,0005$ i $\eta_{100} = 0,0001$. Błąd ten wynosi 10,44% dla zbioru uczącego i 16,44% dla zbioru testującego. Dla porównania, dla metody MLP1 przy takich samych warunkach (tj. ten sam zbiór uczący i testujący, identyczna struktura sieci i limit iteracji) błędy te wynosiły odpowiednio 12,78% oraz 16,83%.

7.4.4. Dobór parametrów dla metody MLP3

W przypadku metody MLP3 (patrz punkt 5.7.3) należy dobrać wartości parametrów: p_i (współczynnik zwiększania wartości η), p_d (współczynnik zmniejszenia wartości η) oraz k (współczynnik dopuszczalnego wzrostu błędu). Parametry te dobrano przy warunkach:

- zbiór uczący: 90 elementów (po 10 na każdą klasę), utworzony na podstawie jednej serii pomiarów dla jednej osoby,
- zbiór testujący: 180 elementów (po 20 na każdą klasę), utworzony na podstawie jednej serii pomiarów dla jednej osoby,
- liczba neuronów w warstwie ukrytej: 40,
- ograniczenie liczby iteracji: 500.

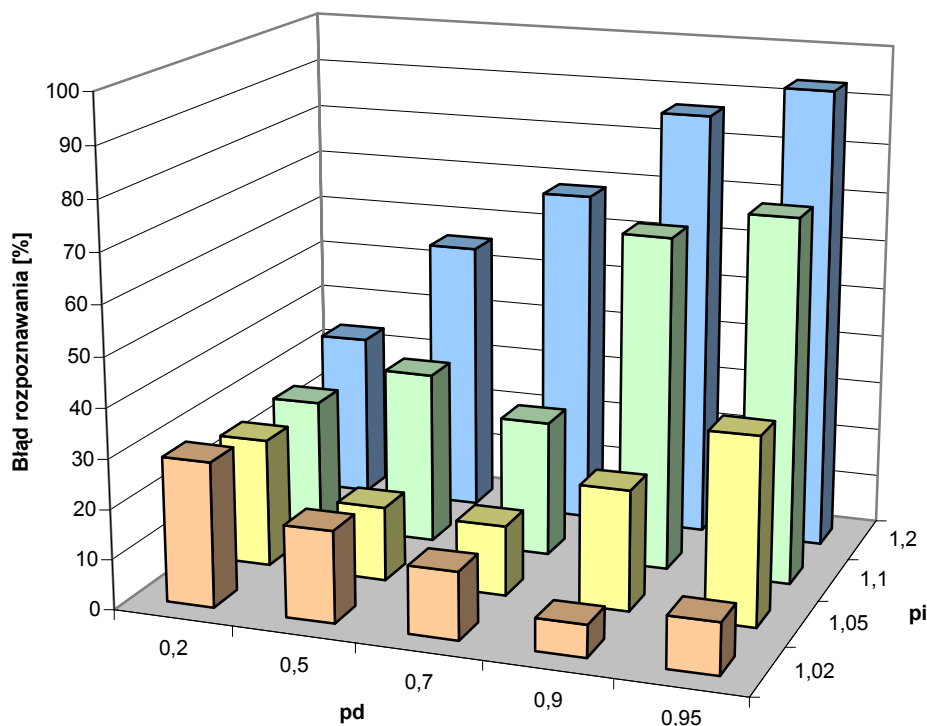
Tabela 7.4.6 zawiera wartości średnie i odchylenia standardowe z 10 pomiarów błędu rozpoznawania dla zbioru uczącego przy różnych wartości współczynników p_i oraz p_d . Tabela 7.4.7 zawiera analogiczne wartości dla zbioru testującego. Wartości najmniejsze wyróżniono. Wyniki dla zbioru uczącego przedstawiono na Rys. 7.4.8 w formie graficznej. Wszystkie pomiary przeprowadzono dla $k = 1,04$.

	p_i							
	1,02		1,05		1,1		1,2	
p_d	EU [%]	DU	EU [%]	DU	EU [%]	DU	EU [%]	DU
0,2	29,11	10,21	25,89	17,05	26,56	10,70	33,56	19,29
0,5	18,44	6,02	14,89	10,12	35,00	24,72	55,44	17,46
0,7	13,56	6,81	14,11	7,96	27,56	18,05	68,44	20,89
0,9	6,44	3,73	24,22	13,92	67,44	11,50	86,78	9,97
0,95	10,33	5,81	37,67	15,83	73,33	10,73	93,33	5,74

Tabela 7.4.6. Wartości średnie błędu rozpoznawania i odchylenia standardowe dla zbioru uczącego w zależności od wartości parametrów p_i i p_d

	p_i							
	1,02		1,05		1,1		1,2	
p_d	ET [%]	DT	ET [%]	DT	ET [%]	DT	ET [%]	DT
0,2	32,67	8,91	30,11	15,48	30,00	9,11	36,50	17,58
0,5	21,94	5,10	20,28	9,43	37,89	22,47	56,39	16,84
0,7	18,39	5,56	19,50	7,47	34,67	18,29	69,33	18,86
0,9	13,06	3,55	28,83	11,51	67,83	11,36	87,00	9,64
0,95	15,78	5,04	40,44	14,64	73,61	10,35	93,33	5,74

Tabela 7.4.7. Wartości średnie błędu rozpoznawania i odchylenia standardowe dla zbioru testującego w zależności od wartości parametrów p_i i p_d

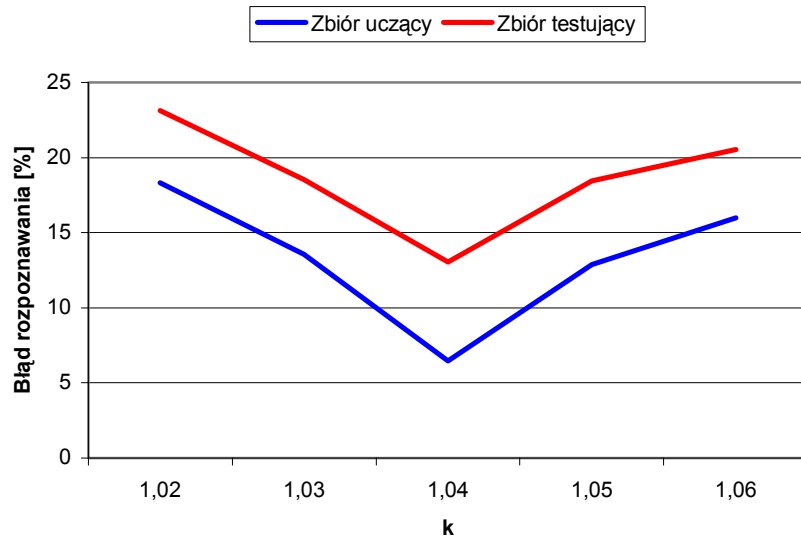


Rys. 7.4.8. Wartości średnie błędu rozpoznawania dla zbioru uczącego w zależności od parametrów p_i oraz p_d przy $k = 1,04$

Najlepsze rezultaty otrzymano dla $p_i = 1,02$ i $p_d = 0,9$. Błąd rozpoznawania dla takich parametrów oraz $k = 1,04$ wynosił 6,44% dla zbioru uczącego i 13,06% dla testującego. Tabela 7.4.8 i Rys. 7.4.9 przedstawiają porównanie średniego błędu rozpoznawania i odchyłeń standardowych dla innych wartości współczynnika dopuszczalnego wzrostu błędu k . Jak widać, wartości inne niż 1,04 powodowały wzrost średniego błędu rozpoznawania.

k	Zbiór uczący		Zbiór testujący	
	EU [%]	DU	ET [%]	DT
1,02	18,33	10,09	23,11	8,64
1,03	13,56	7,16	18,56	5,70
1,04	6,44	3,73	13,06	3,55
1,05	12,89	8,25	18,44	6,87
1,06	16,00	10,35	20,56	9,20

Tabela 7.4.8. Średni błąd rozpoznawania i odchylenie standardowe w zależności od k dla $p_i = 1,02$ oraz $p_d = 0,9$



Rys. 7.4.9. Wykres średniego błędu rozpoznawania w zależności od k dla $p_i = 1,02$ oraz $p_d = 0,9$

Podsumowując, w przypadku metody MLP3 najmniejszy średni błąd rozpoznawania osiągnięto dla parametrów o wartościach: $p_i = 1,02$, $p_d = 0,9$ oraz $k = 1,04$.

7.5. Sieci LVQ

Poniżej przedstawiono wyniki eksperymentów związanych z doбором parametrów i sposobu uczenia sieci LVQ.

7.5.1. Dobór współczynnika uczenia

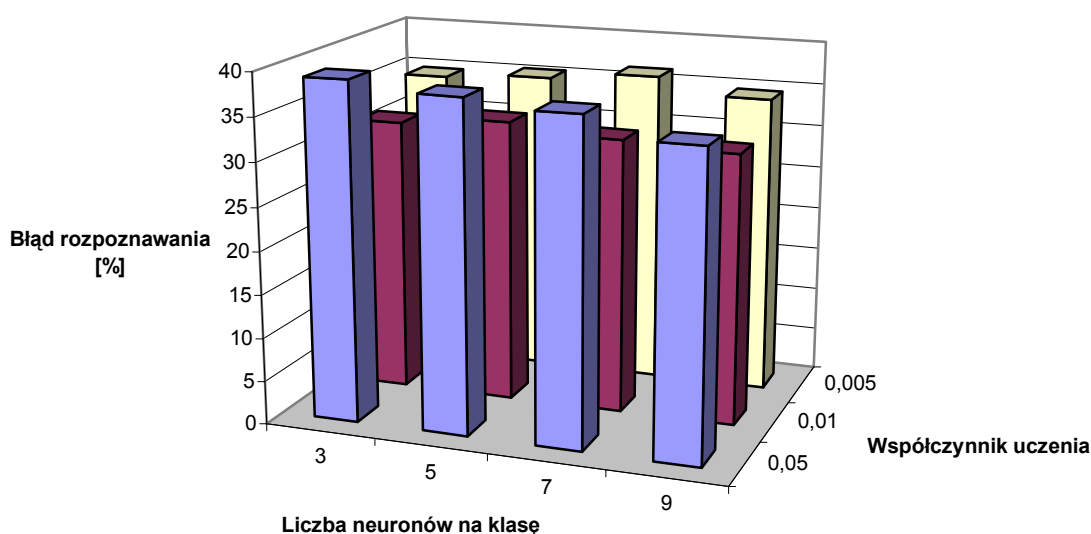
W przypadku sieci LVQ należy określić: ile neuronów sieci przyporządkowanych jest do każdej klasy, jak inicjalizowane są wagi neuronów oraz w jaki sposób sieć jest uczona. W przypadku prostego algorytmu LVQ1 konieczne jest podanie współczynnika uczenia α . Eksperymenty przeprowadzono dla następujących parametrów:

- zbiór uczący: 90 elementów (po 10 na każdą klasę), utworzony na podstawie jednej serii pomiarów dla jednej osoby,
- zbiór testujący: 180 elementów (po 20 na każdą klasę), utworzony na podstawie jednej serii pomiarów dla jednej osoby,
- ograniczenie liczby iteracji: 1000.

Tabela 7.5.1 zawiera średnie i odchylenia standardowe z dziesięciu pomiarów błędu rozpoznawania dla sieci inicjalizowanej w sposób losowy. Pomiary były przeprowadzane dla różnej liczby neuronów (od 3 do 9 na klasę) oraz różnych wartości współczynnika uczenia (0,05, 0,01 oraz 0,005). Najlepsze rezultaty wyróżniono. Wyniki dla zbioru uczącego przedstawiono także w formie graficznej na Rys. 7.5.1.

Liczba neuronów na klasę	Współczynnik uczenia α											
	0,05				0,01				0,005			
	Zbiór uczący		Zbiór testujący		Zbiór uczący		Zbiór testujący		Zbiór uczący		Zbiór testujący	
	EU	DU	ET	DT	EU	DU	ET	DT	EU	DU	ET	DT
3	39,00	0,97	38,67	0,95	31,67	5,47	33,44	6,04	34,67	4,68	35,44	4,18
5	38,00	3,09	37,56	2,39	32,67	5,60	34,28	5,88	35,33	2,08	35,50	2,71
7	37,11	3,64	37,11	3,05	31,67	5,67	33,00	5,72	36,33	3,52	36,39	3,11
9	34,78	3,55	34,94	3,83	31,11	6,50	32,50	6,78	34,56	3,37	35,17	3,35

Tabela 7.5.1. Średni błąd rozpoznawania i odchylenie standardowe przy losowej inicjalizacji wag w zależności od liczby neuronów i współczynnika uczenia

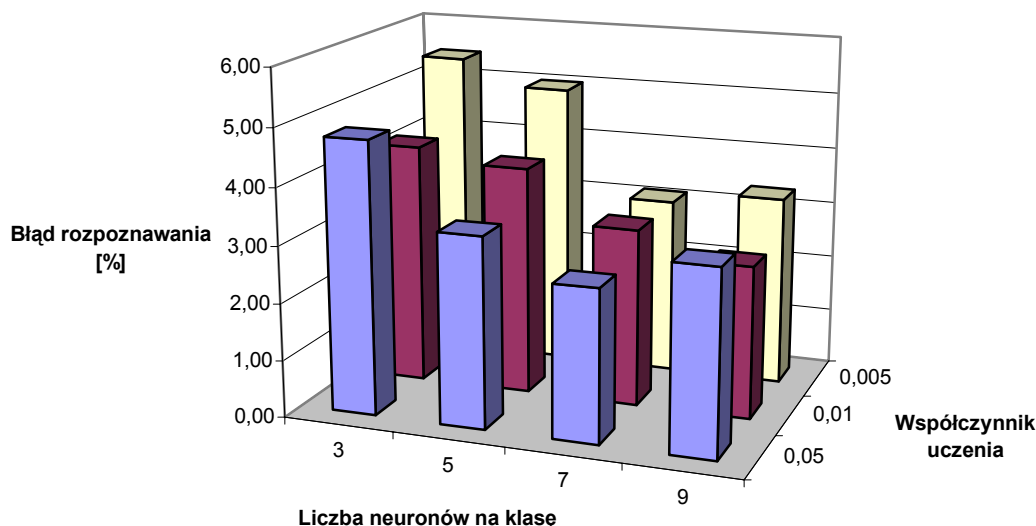


Rys. 7.5.1. Średni błąd rozpoznawania zbioru uczącego przy losowej inicjalizacji wag w zależności od współczynnika uczenia i liczby neuronów

Tabela 7.5.2 zawiera analogiczne wyniki, jednak w tym przypadku inicjalizacja wag nie przebiegała w sposób losowy lecz na podstawie danych ze zbioru uczącego, zgodnie z algorytmem przedstawionym w punkcie 6.5.1. Na Rys. 7.5.2 przedstawiono wyniki rozpoznawania zbioru uczącego.

Liczba neuronów na klasę	Współczynnik uczenia											
	0,05				0,01				0,005			
	Zbiór uczący		Zbiór testujący		Zbiór uczący		Zbiór testujący		Zbiór uczący		Zbiór testujący	
	EU	DU	ET	DT	EU	DU	ET	DT	EU	DU	ET	DT
3	4,78	1,29	6,33	1,53	4,22	1,37	6,28	0,79	5,44	0,97	6,67	1,11
5	3,33	0,91	5,83	0,92	4,00	1,19	5,67	0,94	5,00	0,79	7,00	0,99
7	2,67	0,78	5,67	0,73	3,11	0,88	5,56	1,01	3,11	1,02	5,39	0,83
9	3,22	0,82	5,67	0,73	2,67	0,57	4,83	0,91	3,33	0,52	5,28	0,54

Tabela 7.5.2. Średni błąd rozpoznawania i odchylenie standardowe przy inicjalizacji wag na podstawie zbioru uczącego w zależności od liczby neuronów i współczynnika uczenia



Rys. 7.5.2. Średni błąd rozpoznawania zbioru uczącego przy inicjalizacji wag na podstawie zbioru uczącego w zależności od współczynnika uczenia i liczby neuronów

Otrzymane wyniki pozwalają stwierdzić, że niezależnie od sposobu inicjalizacji wag i liczby neuronów, zastosowanie współczynnika uczenia o wartości 0,01 daje lepsze rezultaty niż zastosowanie $\alpha = 0,05$. Przy mniejszym współczynniku uczenia algorytm był w stanie dokładniej dopasować rozłożenie neuronów. Kolejne zmniejszenie α spowodowało jednak wzrost błędu, co związane jest z ograniczeniem iteracji.

Rodzaj inicjalizacji ma ogromny wpływ na przebieg uczenia. W przypadku inicjalizacji losowej nie udało się osiągnąć błędu mniejszego niż 30%. Inicjalizacja na podstawie przykładów ze zbioru uczącego, dla tej samej liczby iteracji, pozwoliła osiągnąć błąd 2,57% dla zbioru uczącego i 4,85 dla zbioru testującego.

Nie zaobserwowano znaczących różnic w wynikach przy wzroście liczby neuronów w przypadku inicjalizacji losowej. W przypadku inicjalizacji na podstawie zbioru uczącego, wyraźnie widać polepszenie rezultatów przy zwiększaniu liczby neuronów.

7.5.2. Zastosowanie LVQ2

Uczenie sieci LVQ można przeprowadzić w dwóch etapach. Etap pierwszy polega na zastosowaniu algorytmu LVQ1 do wstępnej inicjalizacji wag. W etapie drugim następuje dopasowanie wartości wag przy użyciu algorytmu LVQ2, co ma na celu zmniejszenie błędu rozpoznawania. Poniżej przedstawiono wyniki badań określających wpływ stosowania metody LVQ1 w połączeniu z LVQ2, oznaczonej dalej jako LVQ1+2, na błąd rozpoznawania.

Eksperymenty przeprowadzono dla następujących parametrów:

- zbiór uczący: 90 elementów (po 10 na każdą klasę), utworzony na podstawie jednej serii pomiarów dla jednej osoby,
- zbiór testujący: 180 elementów (po 20 na każdą klasę), utworzony na podstawie jednej serii pomiarów dla jednej osoby,
- ograniczenie liczby iteracji: 500,
- warunek stopu dla etapu wykorzystującego LVQ2: limit iteracji lub brak modyfikacji po zaprezentowaniu całego zbioru uczącego,
- szerokość okna $w = 0,2$,
- losowa inicjalizacja wag początkowych.

Tabela 7.5.3 zawiera średnie z dziesięciu pomiarów błędu rozpoznawania zbioru uczącego w przypadku stosowania metody LVQ1 oraz LVQ1+2. Pomiary przeprowadzono dla różnych liczb neuronów oraz różnych wartości współczynnika uczenia. Tabela 7.5.4 przedstawia analogiczne wyniki dla zbioru testującego. Najlepsze rezultaty wyróżniono.

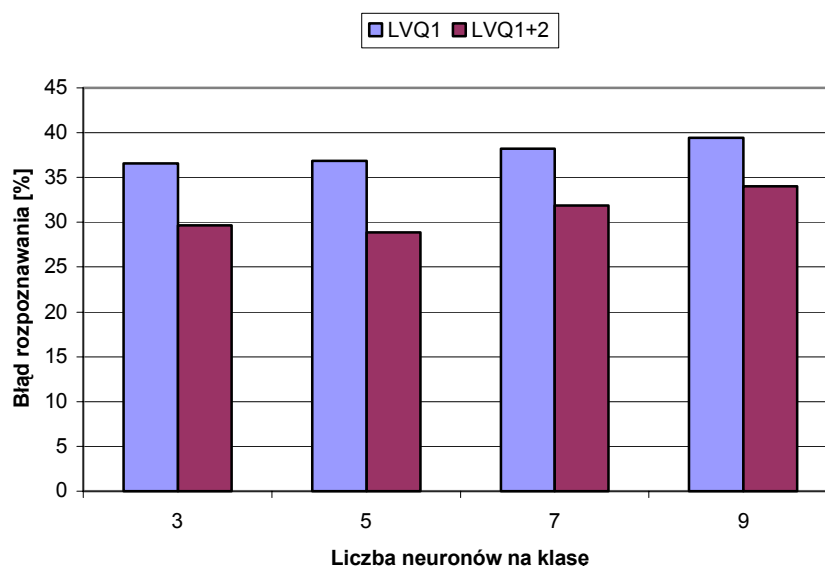
Liczba neuronów na klasę	Współczynnik uczenia							
	0,1		0,05		0,01		0,005	
	LVQ1	LVQ1+2	LVQ1	LVQ1+2	LVQ1	LVQ1+2	LVQ1	LVQ1+2
3	39,00	33,33	38,78	33,33	36,56	29,67	41,11	34,56
5	42,56	34,44	38,89	33,33	36,89	28,89	42,89	35,22
7	42,78	33,33	35,89	30,00	38,22	31,89	42,78	35,33
9	44,56	35,78	35,00	29,11	39,44	34,00	41,67	34,33

Tabela 7.5.3. Średni błąd rozpoznawania zbioru uczącego dla metod LVQ1 oraz LVQ1+2 w zależności od liczby neuronów oraz wartości współczynnika uczenia

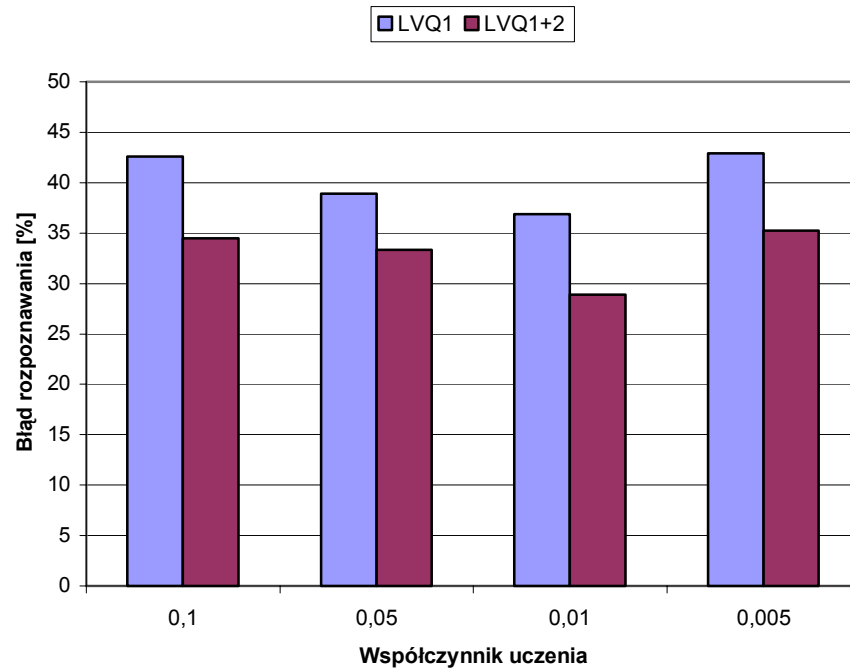
Liczba neuronów na klasę	Współczynnik uczenia							
	0,1		0,05		0,01		0,005	
	LVQ1	LVQ1+2	LVQ1	LVQ1+2	LVQ1	LVQ1+2	LVQ1	LVQ1+2
3	38,00	34,94	38,17	35,00	37,67	32,44	39,94	36,44
5	42,44	36,00	38,00	34,94	37,22	32,00	41,89	36,78
7	43,89	34,78	35,33	31,78	38,72	34,06	41,50	36,94
9	45,22	37,39	34,89	31,00	39,28	35,61	40,06	35,89

Tabela 7.5.4. Średni błąd rozpoznawania zbioru testującego dla metod LVQ1 oraz LVQ1+2 w zależności od liczby neuronów oraz wartości współczynnika uczenia

Na Rys. 7.5.3 oraz Rys. 7.5.4 przedstawiono część wyników w formie wykresów.



Rys. 7.5.3. Średni błąd rozpoznawania zbioru uczącego dla metod LVQ1 oraz LVQ1+2 w zależności od liczby neuronów przy współczynniku uczenia $\alpha = 0,01$



Rys. 7.5.4. Średni błąd rozpoznawania zbioru uczącego dla metod LVQ1 oraz LVQ1+2 w zależności od współczynnika uczenia przy 5 neuronach na klasę

Badania potwierdzają, że zastosowanie algorytmu LVQ2, jako dodatkowego etapu dopasowania wartości wag, może znacząco poprawić rezultaty osiągane przez sieć. Dla losowej inicjalizacji wag osiągnięto średni błąd rozpoznawania mniejszy niż 30% (np. dla $\alpha = 0,01$ oraz 5 neuronów na klasę).

7.5.3. Zastosowanie dopasowania sieci

Algorytm dopasowania sieci (opisany w punkcie 6.5.3) umożliwia modyfikację wag sieci w taki sposób, aby nieużywane neurony dopasowały swoje wagi do najgorzej reprezentowanych przypadków. Poniżej przedstawiono porównanie prostego algorytmu LVQ1 z algorytmem zmodyfikowanym, wykorzystującym dopasowanie sieci, oznaczonym przez 3*LVQ1. Algorytm zmodyfikowany w trzech etapach wykonuje obliczenia według algorytmu LVQ1, ale między etapami następuje dopasowanie nieużywanych neuronów. Dla porównania algorytmów założono, że łączna liczba iteracji wykonywanych przez 3*LVQ1 ma być równa liczbie iteracji wykonywanych przez LVQ1.

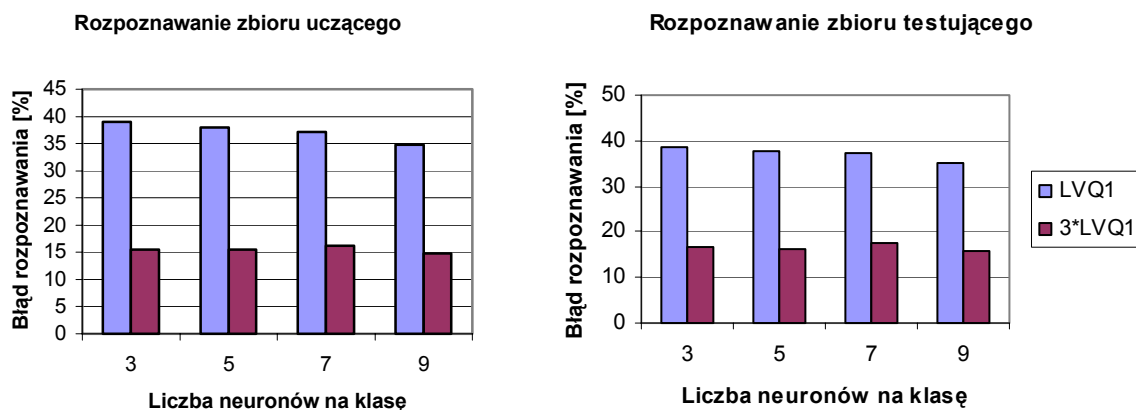
Eksperymenty przeprowadzono dla następujących parametrów:

- zbiór uczący: 90 elementów (po 10 na każdą klasę), utworzony na podstawie jednej serii pomiarów dla jednej osoby,
- zbiór testujący: 180 elementów (po 20 na każdą klasę), utworzony na podstawie jednej serii pomiarów dla jednej osoby,
- ograniczenie liczby iteracji dla LVQ1: 1000,
- ograniczenie liczby iteracji na pojedynczy etap 3*LVQ1: 333,
- współczynnik uczenia: $\alpha = 0,05$.

Tabela 7.5.5 zawiera średnie z 10 pomiarów błędu rozpoznawania sieci uczonych metodami LVQ1 oraz 3*LVQ1, w przypadku losowej inicjalizacji wag sieci. Pomiary przeprowadzono dla różnych ilości neuronów. Wyniki z tabeli przedstawione są także w formie graficznej na Rys. 7.5.5.

Liczba neuronów na klasę	Zbiór uczący		Zbiór testujący	
	LVQ1	3*LVQ1	LVQ1	3*LVQ1
3	39,00	15,56	38,67	16,50
5	38,00	15,56	37,56	16,44
7	37,11	16,22	37,11	17,33
9	34,78	14,78	34,94	15,94

Tabela 7.5.5. Średni błąd rozpoznawania zbioru uczącego i testującego dla sieci uczonych metodami LVQ1 oraz 3*LVQ1 w zależności od liczby neuronów; losowa inicjalizacja wag

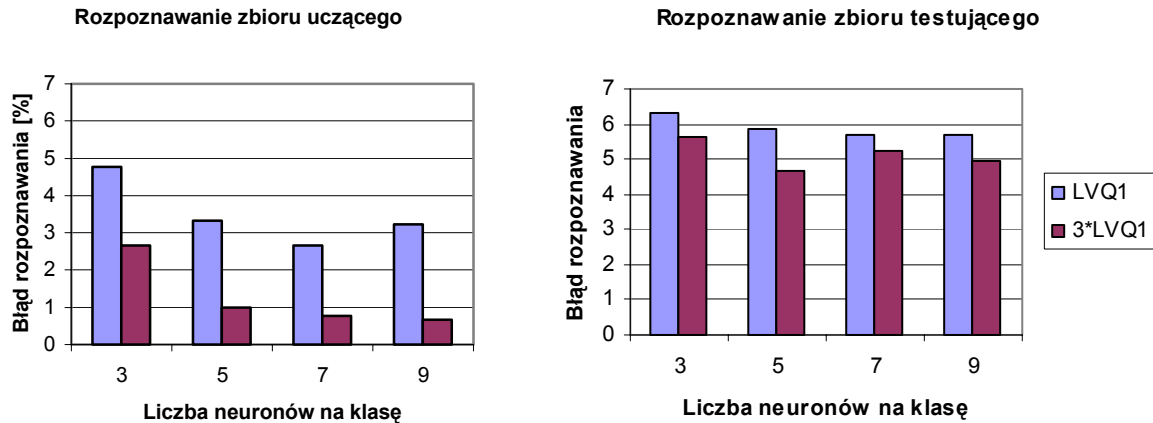


Rys. 7.5.5. Średni błąd rozpoznawania zbioru uczącego i testującego dla sieci uczonych metodami LVQ1 oraz 3*LVQ1 w zależności od liczby neuronów; losowa inicjalizacja wag

Tabela 7.5.6 zawiera analogiczne wyniki dla przypadku, gdy wartości wag inicjalizowane są na podstawie przykładów ze zbioru uczącego. Graficznie wyniki przedstawione są na Rys. 7.5.6.

Liczba neuronów na klasę	Zbiór uczący		Zbiór testujący	
	LVQ1	3*LVQ1	LVQ1	3*LVQ1
3	4,78	2,67	6,33	5,61
5	3,33	1,00	5,83	4,67
7	2,67	0,78	5,67	5,22
9	3,22	0,67	5,67	4,94

Tabela 7.5.6. Średni błąd rozpoznawania zbioru uczącego i testującego dla sieci uczonych metodami LVQ1 oraz 3*LVQ1 w zależności od liczby neuronów; inicjalizacja wag na podstawie zbioru uczącego



Rys. 7.5.6. Średni błąd rozpoznawania zbioru uczącego i testującego dla sieci uczonych metodami LVQ1 oraz 3*LVQ1 w zależności od liczby neuronów; inicjalizacja wag na podstawie zbioru uczącego

Powyższe wyniki pozwalają stwierdzić, że metoda wykorzystująca dopasowanie sieci jest bardzo efektywna. W przypadku losowej inicjalizacji wag błąd rozpoznawania spada do wartości około 15%. W przypadku inicjalizacji wag na podstawie zbioru uczącego błąd na zbiorze uczącym spadł poniżej 1% (dla dużej liczby neuronów), a na zbiorze testującym osiągnął wartość 4,67%. Należy zauważyć, że przy stosowaniu tej metody duża liczba neuronów ułatwia dopasowanie się sieci do zbioru uczącego, ale nie powoduje widocznego spadku błędu rozpoznawania zbioru testującego.

7.6. Porównanie metod klasyfikacji

7.7. Podsumowanie

Literatura

- [1] Stanisław Osowski, „Sieci neuronowe w ujęciu algorytmicznym”, Wydawnictwa Naukowo-Techniczne, Warszawa, 1996
- [2] Daisuke Nishikawa, Yu Wenwei, Hiroshi Yokoi, Yukinori Kakazu, „EMG Prosthetic Hand Controller using Real-time Learning Method”, The 1999 IEEE Systems, Man, and Cybernetics Conference, Tokyo, 1999
- [3] Timothy Masters, „Sieci neuronowe w praktyce, programowanie w języku C++”, Wydawnictwa Naukowo-Techniczne, Warszawa, 1996
- [4] Jari A. Kangas, Teuvo K. Kohonen, Jorma T. Laaksonen, „Variants of Self-Organizing Maps“, IEEE Transactions on Neural Networks, Vol. 1, No. 1, March 1990
- [5] Teuvo K. Kohonen, „Self-Organizing Maps”, Springer, Berlin, 1995
- [6] Marek Kurzyński, „Rozpoznawanie obiektów, metody statystyczne“, Oficyna Wydawnicza PWR, Wrocław, 1997
- [7] Carlo J. De Luca, „The Use of Surface Electromyography in Biomechanics”, Delsys Incorporated, 1997
- [8] Carlo J. De Luca, „Surface Electromyography, Detection and Recording”, Delsys Incorporated, 1996
- [9] Jerzy A. Moczko, Lucyna Kramer, „Cyfrowe metody przetwarzania sygnałów biomedycznych”, Wydawnictwo Naukowe, Poznań, 2001
- [10] Michale R. Neuman, „Biopotential Electrodes”, „The Biomedical Engineering Handbook, Second Edition”, CRC Press LLC, 2000
- [11] Joachim H. Nagel, „Biopotential Amplifiers”, „The Biomedical Engineering Handbook, Second Edition”, CRC Press LLC, 2000
- [12] „INA116, Instrumentation Amplifier”, Burr-Brown, 1994
<http://www-s.ti.com/sc/ds/ina116.pdf>
- [13] Jouko Lampinen, Jorma Laaksonen, Erkki Oja, „Neural Network Systems, Techniques nad Applications in Pattern Recognition”, Otaniemi, 1997
http://zeus.hut.fi/publications/ps/b1_nnsystems.ps
- [14] Tomasz Białobrzeg, „Analiza pola miopotencjałów rejestrowanych z mięśni przedramienia człowieka w sterowaniu bioprotezą”, Praca magisterska, Politechnika Wrocławska, Wrocław, 2003
- [15] Strona internetowa firmy Otto Bock, www.ottobock.com
- [16] Krzysztof Krzysztoforski, „System automatycznej detekcji sygnałów z mięśnia ludzkiego oparty na μK MC68HC11A1”, Praca magisterska, Politechnika Wrocławska, Wrocław, 2000
- [17] Strona internetowa Laboratory of Autonomous Systems Engineering, Hokkaido University,
<http://junji.complex.eng.hokudai.ac.jp/projects/ProstheticHand/cgi/sender.cgi?index>
- [18]

Dodatek A – Opis programu ArtificialHand

Dodatek B - Opis programu EMGRecorder

Dodatek C – Implementacja wybranych metod