POLITECHNIKA WROCŁAWSKA

WYDZIAŁ ELEKTRONIKI

PRACA

DOKTORSKA

Zastosowanie sieci neuronowych typu SVM

do

rozpoznawania mowy

AUTOR:

mgr inż. Piotr Walendowski

PROMOTOR:

prof. dr hab. inż. Marian Piekarski,

Zakład Teorii Obwodów

Instytut Telekomunikacji,
Teleinformatyki i Akustyki

OCENA PRACY:

Słowa kluczowe:

rozpoznawanie izolowanych słów, wybór początku i końca, rozpoznawanie mowy, technika wektorów podtrzymujących,

endpoint detection, Voice Activity Detection, VAD, speech recognition, Support Vector Machine, SVM, one class SVM.

WROCŁAW 2008

Nazywam się Piotr Walendowski. Urodziłem się 11. kwietnia 1978 roku we Wrocławiu. Początkowo uczęszczałem do Państwowej Podstawowej Szkoły Muzycznej im. Karola Szymanowskiego we Wrocławiu do sekcji smyczkowej. Od trzeciej klasy uczyłem się w Szkole Podstawowej nr 1 im. Marii Dąbrowskiej, którą ukończyłem z wyróżnieniem. Naukę kontynuowałem w XIV. Liceum Ogólnokształcącym im. Polonii Belgijskiej w klasie ogólnej z poszerzonym językiem angielskim. Jako uczelnię wyższą wybrałem Politechnikę Wrocławską, gdzie studiowałem na Wydziale Elektroniki na kierunku Elektronika i Telekomunikacja. Na późniejszych latach studiów wybrałem kolejno, jako specjalizację – Telekomunikację, a później jako profil dyplomowania – Systemy cyfrowe. Pracę dyplomową „Wybieranie numerów telefonicznych hasłem wypowiadanym przez człowieka" zrealizowałem w Instytucie Telekomunikacji i Akustyki PWr w Zakładzie Teorii Obwodów. Studia doktoranckie kontynuowałem na macierzystej uczelni także na wydziale Elektroniki w Instytucie Telekomunikacji, Teleinformatyki i Akustyki Politechniki Wrocławskiej.

Pracę tę dedykuję mojej Żonie - Agacie oraz mojemu Synowi - Michałowi. To oni są dla mnie źródłem siły, dzięki której możliwe stało się napisanie tej rozprawy.

Chcę także gorąco podziękować promotorowi tej pracy - prof. dr hab. inż. Marianowi Piekarskiemu. Bez poświęcenia wielu godzin na konsultacje oraz bez Jego wielu cennych uwag i wskazówek, praca ta nie mogłaby powstać w takiej formie, jaką czytelnik ma przed sobą.

Dziękuję wszystkim, którzy wspierali mnie swoim słowem i postawą w czasie prac nad rozprawą, a w szczególności moim Rodzicom i Przyjaciołom.

Wykaz skrótów użytych w pracy

W pracy przyjęto zasadę stosowania skrótów angielskich ze względu na fakt, że wiele z używanych oznaczeń nie ma jeszcze przyjętych skrótów w języku polskim lub też skróty angielskie są bardziej popularne.

A/D (ang. *Analog/Digital*) – analogowo/cyfrowy (A/C)

AI (ang. *Artificial Intelligence*) – sztuczna inteligencja

ANN (ang. *Artificial Neural Network*) – sztuczna sieć neuronowa (SSN)

*ARC (*ang. *Arcsine Reflection Coefficients)* – inna nazwa na parametry ISP (ang. *Inverse Sine Parameters*)

ASR (ang. *Automatic Speech Recognition*) – automatyczne rozpoznawanie mowy (ARM)

BP (ang. *Background Propagation*) – propagacja wsteczna

CPLD (ang. *Constant Positive Linear Dependance Qualification*) - warunek stałej, dodatniej, liniowej zależności w warunkach Karush-Kuhn-Tucker'a

CRCQ (ang. *Constant Rank Constraint Qualification*) – warunek stałego rzędu w warunkach Karush-Kuhn-Tucker'a

CVGS (ang. *Cross-Validation with Grid-Search*) – skrót określający użycie techniki *grid-search* poprzedzonej metodą *cross-validation*.

DAGSVM (ang. *Directed Acylic Graph Support Vector Machines*) – jeden z algorytmów wielokrotnej klasyfikacji w sieciach SVM

DARPA (ang. *Defence Advanced Research Projects Agency*) – amerykańska agencja ds. zaawansowanych obronnych projektów badawczych

DP (ang. *Dynamic Programming*) – programowanie dynamiczne

DTW (ang. *Dynamic Time Warping*) – dynamiczne zwijanie czasowe

FIR (ang. *Finite Input Response*) – filtr o skończonej odpowiedzi impulsowej (SOI)

HMM (ang. *Hidden Markov Model*) – ukryte modele Markowa

IMS (ang. *Interactive Multimodal System*) – interaktywny system multimodalny

ISP (ang. *Inverse Sine Parametrs*) – parametry odwrotnej funkcji sinus

ITU (ang. *International Telecommunication Union*) – Międzynarodowa Unia Telekomunikacyjna

IZCT (ang. *sIlence Zero Crossing Threshold*) – próg liczby przejść przez zero dla szumu tła

KKT (ang. *Karush-Kuhn-Tucker conditions*) – warunki Karush-Kuhn-Tucker'a

LAR (ang. *Log Area Ratio Parameters*) – współczynniki logarytmicznego stosunku powierzchni tuby akustycznej $\log\ g_i$

LICQ (ang. *Linear Independence Constraint Qualification*) – warunek liniowej niezależności w warunkach Karush-Kuhn-Tucker'a

LPC (ang. *Linear Predictive Coding*) – liniowe kodowanie predykcyjne

LPCC (ang. *Linear Predictive Cepstrum Coding Parametres*) – współczynniki liniowego kodowania predykcyjnego cepstrum

LSF - (ang. *Line Spectral Frequencies)* – częstotliwości widma liniowego

LSP - (ang. *Line Spectrum Pair)* – inna nazwa na LSF (ang. *Line Spectral Frequencies*)

LVQN (ang. *Learning Vector Quantization Network*) – typ sieci neuronowej wykorzystującej kwantowanie wektorowe

MFCC (ang. *Mel Frequency Cepstral Coefficients*) – cepstralne współczynniki melowe

MFCQ (ang. *Mangasarian-Fromowitz constraint qualification*) – warunek Mangasarian'a-Fromowitz'a w warunkach Karush-Kuhn-Tucker'a

MIT (ang. *Massachusetts Institute of Technology Lincoln Labs*) – laboratoria Lincolna Instytutu Technologii Massachusetts

MLP (ang. *Multilayer Perceptron*) – perceptron wielowarstwowy

MRSED (ang. *Modified Rabiner-Sambur Endpoint Detection algorithm*) – zmodyfikowany przez autora algorytm Rabinera-Sambura wyboru początku i końca słowa

NPC (ang. *Neural Predictive Coding*) – neuronowe kodowanie predykcyjne

PARCOR (ang. *PARtial CORrelation coefficients*) – metoda częściowych współczynników korelacji

PLP (ang. *Perceptual Linear Prediction*) – metoda parametryzacji wykorzystująca perceptualną predykcję liniową

RASTA (ang. *RelAtive SpecTrA*) – metoda parametryzacji wykorzystująca widmo względne

RASTA-PLP – metoda parametryzacji łącząca cechy metody PLP i RASTA

RC (ang. *Reflection Coefficients*) – współczynniki odbicia k_i

SLS (ang. *Spoken Language System - SLS*) – system języka mówionego

SMO (ang. *Sequential Minimal Optimization Algorithm*) – algorytm minimalnej optymalizacji sekwencyjnej

SNR (ang. *Signal to Noise Ratio*) – współczynnik sygnał – szum

SVM (ang. *Support Vector Machines*) – technika wektorów podtrzymujących

TDNN (ang. *Time Delay Neural Network*) – sieć neuronowa z opóźnieniem czasowym

VAD (ang. *Voice Activity Detection*) – wybór początku i końca wypowiedzi (ang. *end-point location*)

VAD-1SVM – autorski system wyboru początku i końca słowa z użyciem sieci One Class SVM

VQ (ang. *Vector Quantization*) – kwantowanie wektorowe

Spis treści

Wykaz skrótów użytych w pracy 3
Spis treści 5
1. Wstęp 8
1.1 Rys historyczny rozwoju systemów automatycznego rozpoznawania mowy 8
1.2 Teza, cele i założenia rozprawy 10
1.3 Układ pracy 11
2. Przegląd metod automatycznego rozpoznawania mowy 12
2.1 Etap przetwarzania wstępnego sygnału mowy 13
2.1.1 Akwizycja sygnału mowy 13
2.1.2 Preemfaza 14
2.1.3 Wybór początku i końca słowa 15
2.1.4 Segmentacja 16
2.1.5 Okienkowanie 17
2.2 Etap parametryzacji sygnału mowy 19
2.2.1 Parametry czasowe 19
2.2.2 Parametry częstotliwościowe 20
2.2.3 Cepstrum 20
2.2.4 Perceptualne parametry cepstralne 22
2.2.5 Liniowe Kodowanie Predykcyjne 24
2.2.6 Liniowe kodowanie predykcyjne cepstrum 28
2.2.7 Alternatywne metody reprezentacji współczynników predykcji 29
2.3 Etap klasyfikacji 33
2.3.1 Podział systemów automatycznego rozpoznawania obrazów 33
2.3.2 Algorytmy rozpoznawania 34
2.3.3 Funkcja podobieństwa 35
2.3.4 Inne metody klasyfikacji 36
2.3.5 Sztuczne sieci neuronowe 38
3. Sieci SVM 40
3.1 Wstęp do techniki wektorów podtrzymujących 40
3.2 Rozwiązanie problemu klasyfikacji wzorców liniowo separowanych z użyciem sieci SVM 42

3.3 Rozwiązanie problemu klasyfikacji wzorców liniowo nieseparowalnych z użyciem liniowej sieci SVM 46
3.4 Struktura nieliniowej sieci SVM 47
3.5 Własności nieliniowej sieci SVM 49
3.6 Sieć typu One-class SVM 50
3.7 Rozpoznawanie wielu klas w sieci SVM 51
3.7.1 Algorytm typu „jeden przeciw wszystkim” 52
3.7.2 Algorytm typu „jeden przeciw jednemu” 53
3.7.3 Inne algorytmy wielokrotnej klasyfikacji 54
3.8 Optymalizacja sieci SVM 54
4. Projekt i testy systemu rozpoznawania izolowanych słów z użyciem sieci typu SVM 56
4.1 Założenia projektowanego systemu rozpoznawania izolowanych słów 56
4.2 Wyposażenie sprzętowe i programowe stanowiska laboratoryjnego 57
4.3 Bazy nagrań sygnałów mowy 57
4.3.1 Własna pierwotna baza wypowiedzi 58
4.3.2 Własna Baza Imion 58
4.3.3 Baza CORPORA 58
4.4 Opis testów w środowisku zaszumionym 60
4.4.1 Stosunek sygnału do szumu 60
4.4.2 Wybór typu szumów do testów 62
4.5 Opracowanie i testy nowego algorytmu wyboru początku i końca wypowiedzi z użyciem sieci typu One Class SVM. 63
4.5.1 Zmodyfikowany algorytm L. R. Rabinera i M. R. Sambura (MRSED) 64
4.5.2 Wstępne testy algorytmu MRSED w środowisku zaszumionym oraz wpływ filtru preemfazy na skuteczność działania algorytmu 68
4.5.3 Zastosowanie parametrów z rekomendacji ITU-T G729B 74
4.5.4 Wpływ filtru preemfazy na parametry ZCR 76
4.5.5 Wpływ operacji podniesienia do kwadratu parametrów ΔZCR na reprezentację sygnału mowy 77
4.5.6 Możliwość zastosowania parametrów VAD na etapie parametryzacji 78
4.5.7 Algorytm VAD oparty na sieciach SVM 79
4.5.8 Porównanie zmodyfikowanego algorytmu Rabinera-Sambura z algorytmem VAD-1SVM 86
4.5.9 Interpretacja wyników testów i podsumowanie badań nad algorytmami wyboru początku i końca słowa 92
4.6 Zastosowanie sieci typu SVM do klasyfikacji i końcowe testy systemu 93

4.6.1 Wyjaśnienie oznaczeń stosowanych w opisach testów 94
4.6.2 Ustawienia systemu stosowane w testach..... 95
4.6.3 Pomiar skuteczności rozpoznawania i poziom pewności pomiarów 97
4.6.4 Wpływ wyboru metody parametryzacji na wyniki rozpoznawania w przypadku wielu mówców i braku algorytmu VAD 98
4.6.5 Wpływ wyboru metody parametryzacji na wyniki rozpoznawania w przypadku jednego mówcy i zastosowania algorytmu VAD 100
4.6.6 Wpływ długości wektorów na wyniki rozpoznawania..... 102
4.6.7 Porównanie metody uzupełniania wektorów zerami z metodą stałej liczby próbek 105
4.6.8 Wpływ wartości *C* i *γ* na wyniki rozpoznawania oraz liczbę wektorów podtrzymujących 107
4.6.9 Wpływ skalowania i CVGS na wyniki rozpoznawania..... 109
4.6.10 Wpływ wyboru typu jądra w sieci SVM na wyniki rozpoznawania..... 110
4.6.11 Wpływ wielkości słownika na wyniki rozpoznawania..... 111
4.6.12 Wpływ liczby powtórzeń na etapie uczenia sieci SVM na wyniki rozpoznawania 113
4.6.13 Wpływ wyboru algorytmu VAD na skuteczność rozpoznawania..... 115
4.6.14 Badanie skuteczności systemu dla wielu mówców..... 117
4.6.15 Porównanie skuteczności systemu rozpoznawania wykorzystującego SVM z systemem opartym na ukrytych modelach Markowa 118
4.7 Podsumowanie eksperymentów i ostateczny opis parametrów systemu rozpoznawania mowy z użyciem sieci SVM 121
5. Zakończenie..... 125
5.1 Dalsze możliwości rozwinięcia systemu 125
5.2 Podsumowanie wyników rozprawy..... 126
Dodatek A – Problem poszukiwania ekstremów metodą mnożników Lagrange'a..... 128
Dodatek B – Zawartość płyty CD-ROM..... 131
Bibliografia 132
Spis ilustracji..... 140
Spis tabel 142

1. Wstęp

1.1 Rys historyczny rozwoju systemów automatycznego rozpoznawania mowy

Idea automatycznego rozpoznawania mowy (ang. *Automatic Speech Recogniton – ASR*) nie jest nowa. Pierwsze próby sięgają lat 50-tych XX w., kiedy Davis, Biddulpth i Balashek w laboratoriach Bella zbudowali, z wykorzystaniem parametrów widmowych, system rozpoznający dziesięć cyfr wypowiadanych przez jednego mówcę [1]. Niezależne prace w tej dziedzinie rozpoczęły także laboratoria RCA (Olson i Belar) [2] oraz MIT (Forgie i Forgie) [3].

W latach 60-tych ubiegłego wieku zaczęły powstawać pierwsze sprzętowe realizacje systemów rozpoznawania mowy. To okres, w którym na arenę międzynarodową w tej dyscyplinie weszli Japończycy z Radio Research Lab w Tokyo (Suzuki i Nakata) [4] oraz laboratoria NEC (Nagata).

W tym też okresie w ówczesnym Związku Radzieckim Vintsyuk [5] zaproponował zastosowanie programowania dynamicznego (ang. *Dynammic Programming - DP*) do porównywania wzorców o różnej długości.

Lata 70-te to kolejny rozwój systemów ASR. W tym okresie Itakura [6] zaadoptował liniowe kodowanie predykcyjne, stosowane dotąd w kodowaniu, do tworzenia efektywnych systemów rozpoznawania mowy. Pod koniec tej dekady w laboratoriach AT&T Bella [7] rozpoczęto prace nad systemem rozpoznawania mowy niezależnym od mówcy. Badania te stały się podstawą do dalszych prac nad systemami tego typu, które prowadzone są do chwili obecnej.

Lata 80-te zostały zdominowane przez metody statystycznego modelowania, a w szczególności przez ukryte modele Markowa (ang. *Hidden Markov Models – HMM*) [8]. Mimo że metoda ta była już wcześniej znana, to dopiero w tym okresie stała się popularna w zastosowaniu do systemów ASR. Także pod koniec tej dekady wzrosło na nowo zainteresowanie sztucznymi sieciami neuronowym (ang. *Artificial Neural Networks – ANN*) w zastosowaniu do rozpoznawania mowy [9], [10].

Koniec lat 80-tych i początek 90-tych to okres prac nad systemami rozpoznawania mowy ciągłej z dużym słownikiem. Projekty tego typu zainicjowała agencja DARPA (ang. *Defence Advanced Research Projects Agency*) z celem stworzenia wysokiej wydajności systemu ASR dla mowy ciągłej o pojemności 1000 słów. Prace nad tego typu systemami prowadzono także m.in. w laboratoriach AT&T Bella [11].

Od lat 90-tych prowadzi się intensywne prace nad tzw. systemami języka mówionego (ang. *Spoken Language Systems - SLS*) [12]. Ich celem (w uproszczeniu) jest zrozumienie mówcy oraz udzielenie odpowiedniego komunikatu zwrotnego. Szacuje się, że rozwój tej technologii przyczyni się do dalszego rozwoju i dostępności technologii komputerowych, pomoże w komunikacji międzynarodowej, przyczyni się do rozwoju handlu i stworzy wiele nowych miejsc pracy w wielu związanych z tą technologią branżach. Systemy tego typu rozwijane są w kilku kierunkach, z uwzględnieniem takich zagadnień jak:

- zwiększenie odporności na zakłócenia,
- możliwość automatycznego treningu i adaptacji do zmian,
- wsparcie dla mowy ciągłej,
- tworzenie modeli dialogowych, aby w zrozumiały sposób można było prowadzić dialog w obie strony,
- taka generacja odpowiedzi, aby ta była w miarę możliwości naturalna,
- generacja i synteza sygnału mowy,
- systemy wielojęzykowe (włącznie z systemami typu *speech-to-speech*),
- interaktywne systemy multimodalne (ang. *Interactive Multimodal Systems - IMS*) [13][1], aby zwiększyć dokładność i naturalność komunikacji człowiek – komputer poprzez integrację mowy z innymi źródłami informacji, takimi jak mimika twarzy, gesty czy ruchy rąk.

Zastosowań systemów ASR jest wiele. Jednak w ocenie autora ich głównym, choć wciąż odległym celem, jest osiągnięcie takiego poziomu, aby możliwe było rozumienie przez maszynę tego, co człowiek mówi. Umożliwi to dwustronną, skuteczną komunikację maszyna – człowiek z wykorzystaniem głosu przez obie strony jako najbardziej naturalnej formy komunikacji dla człowieka.

W ogólności systemy ASR można podzielić na rozpoznające mówców oraz rozpoznające treść wypowiedzi. Te ostatnie będą tematem dalszych rozważań.

Do zastosowań systemów rozpoznawania treści wypowiedzi należy np. kontrola dostępu na podstawie hasła głosowego, która może być połączona z rozpoznawaniem mówcy. Systemy tego typu z powodzeniem mogą być stosowane także w wywiadzie i innych służbach w zautomatyzowanym podsłuchu telefonicznym. Taki system mógłby być zainstalowany w centrali telefonicznej i np. automatycznie włączać proces rejestracji lub wyszukiwania połączeń, w których zostały użyte określone słowa lub zwroty (np. bomba, zapalnik, itp.). Aspekt ten niewątpliwie nabrał na znaczeniu po zamachach na wielką skalę. Kolejnym bardzo ciekawym zastosowaniem tego typu systemów są automatyczne systemy stenotypujące typu *speech-to-text*, zapisujące przebieg różnego typu spotkań, konferencji, nagrań radiowych czy telewizyjnych. Można je zastosować np. w sądach do rejestracji treści zeznań i przebiegu rozpraw.

Systemy ASR nastawione na rozpoznawanie izolowanych słów (komend) idealnie nadają się do sterowania głosem komputerów, robotów i innych maszyn. Za pomocą etykiet głosowych można także z powodzeniem wybierać adresy zestawianych połączeń telefonicznych czy wideokonferencyjnych lub chociażby adresy poczty elektronicznej. Systemy tego typu są szczególną pomocą dla osób niepełnosprawnych, dzięki ułatwieniu im komunikacji z otoczeniem.

[1] hasło: multimodal interaction - w ten sposób oznaczane będą dalej hasła z Wikipedii angielskiej [13] lub polskiej [107])

1.2 Teza, cele i założenia rozprawy

Teza rozprawy

Możliwe jest zaprojektowanie skutecznego systemu rozpoznawania izolowanych słów, z wykorzystaniem sztucznych sieci neuronowych typu SVM, działającego także w środowisku zaszumionym.

Cele rozprawy

Uwzględniając tezę rozprawy ustalono cel badawczy pracy.

Celem podstawowym rozprawy jest weryfikacja tezy, czyli stwierdzenie, czy istnieje możliwość zaprojektowania skutecznie działającego systemu rozpoznawania izolowanych słów z wykorzystaniem sztucznych sieci neuronowych typu SVM, z możliwością pracy także w środowisku zaszumionym.

Osiągnięcie celu podstawowego wymaga, m.in.:

- właściwego doboru wszystkich elementów systemu rozpoznawania mowy,
- stworzenia algorytmu wyboru początku i końca słowa zdolnego do pracy także w środowisku zaszumionym,
- zaprojektowania systemu rozpoznawania mowy z wykorzystaniem Sztucznych Sieci Neuronowych typu SVM,
- komputerowej symulacji działania systemu,
- wykonania eksperymentów skuteczności pracy systemu,
- opracowania i analizy uzyskanych wyników eksperymentów.

Podstawowe założenia rozprawy

Projektowany system rozpoznawania mowy powinien z wysoką skutecznością rozpoznawać izolowane słowo z grupy innych kilkunastu do kilkudziesięciu słów przy niskim poziomie zewnętrznego szumu (SNR > 32dB), przy założeniu, że z systemu będzie korzystać jeden mówca.

System należy wyposażyć we własny algorytm wyboru początku i końca słowa. Cały system powinien być zdolny do działania także w środowisku zaszumionym.

Dodatkową zaletą projektowanego systemu byłaby możliwość korzystania z niego przez wielu różnych użytkowników. Oznaczałoby to, że system mógłby działać, z danymi pochodzącymi od wielu różnych mówców, zarówno na etapie uczenia jak i dalszej zwykłej jego pracy.

1.3 Układ pracy

Pracę podzielono na kilka głównych części. W rozdziale 1. oprócz wstępu przedstawiono tezę rozprawy, jej cele oraz ogólny układ pracy.

Rozdział 2. jest teoretycznym przedstawieniem technik związanych z rozpoznawaniem mowy. Szczególną uwagę poświęcono metodom wykorzystywanym w dalszej części na etapie projektu i testów systemu.

Rozdział 3 w całości poświęcono technice wektorów podtrzymujących SVM, która jest stosunkowo nowym typem sztucznych sieci neuronowych. Opisano zastosowanie SVM do klasyfikacji wzorców liniowo separowalnych oraz wzorców liniowo nieseparowalnych. W rozdziale przedstawiono ponadto techniki pozwalające zastosować sieci SVM do rozpoznawania wielu klas.

Rozdział 4. zawiera projekt systemu rozpoznawania izolowanych słów, wykorzystujący sieci SVM w dwóch miejscach systemu: w algorytmie wyboru początku i końca słowa oraz na etapie klasyfikacji. W rozdziale opisano doświadczenia, na podstawie których powstały końcowe założenia dotyczące parametrów systemu i algorytmów jego działania. W części tej można również znaleźć wyniki doświadczeń skuteczności rozpoznawania dla zaproponowanego systemu ASR.

W rozdziale 5. zamieszczono wnioski z przeprowadzonych badań, podsumowanie końcowych wyników oraz uzasadnienie prawdziwości tezy rozprawy. Ponadto w rozdziale przedstawiono wskazówki dotyczące potencjalnych dalszych kierunków badań i możliwości rozwinięcia systemu.

Pracę uzupełnia Dodatek A, w którym wyjaśniono problem poszukiwania ekstremów metodą mnożników Lagrange'a oraz Dodatek B w postaci płyty CD-ROM, na której zamieszczono wybrane oprogramowanie i sygnały mowy do dalszych badań.

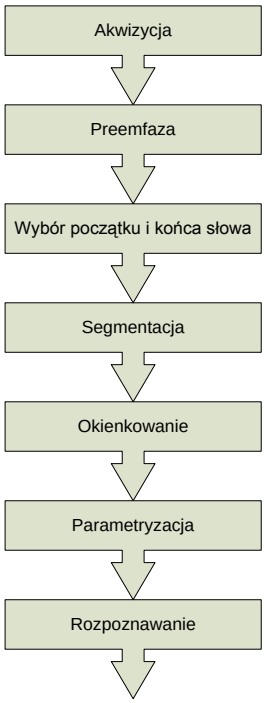
Na końcu pracy zamieszczono bibliografię oraz spis ilustracji i tabel.

Autor uważa rozdziały 4. i 5. za najważniejsze dla całej rozprawy. To one właśnie stanowią jej istotę.

2. Przegląd metod automatycznego rozpoznawania mowy[2]

Aby wypowiedź człowieka mogła zostać „zrozumiana” przez maszynę spośród innych wypowiedzi, należy poddać ją procesowi rozpoznawania mowy.

Ogólny schemat blokowy systemu rozpoznawania mowy przedstawiono na Rys. 2.1



Rys. 2.1 Schemat blokowy systemu rozpoznawania mowy

Pierwszy etap tego procesu nosi nazwę **przetwarzania wstępnego**. Analogowy sygnał mowy zarejestrowany mikrofonem zostaje wzmocniony i poddany filtracji filtrem dolnoprzepustowym. Możliwe jest teraz przekształcenie sygnału analogowego na postać cyfrową z użyciem przetwornika analogowo-cyfrowego A/D (ang. *Analog/Digital*). Następnie sygnał cyfrowy poddawany jest filtracji filtrem preemfazy, w celu uwydatnienia składowych wyższych częstotliwości. Kolejny krok polega na znalezieniu początku i końca wypowiedzianego słowa i odrzuceniu fragmentów leżących poza nimi. Następnie sygnał dzielony jest na mniejsze segmenty zwane ramkami, aby mógł być traktowany jako kwazistacjonarny w obrębie każdej ramki. Jest to szczególnie istotne np. w procesie parametryzacji z użyciem metod liniowego kodowania predykcyjnego LPC (ang. *Linear Predictive Coding*). Sygnały w ramkach mnożone są w kolejnym kroku przez funkcję okna w celu minimalizacji „nierównomierności” występujących na początku i końcu ramki.

Kolejny etap przetwarzania nosi nazwę **parametryzacji sygnału**. Jego istotą jest wydobycie z ramek sygnału takich jego cech charakterystycznych, które będą go najlepiej opisywać. Ilość parametrów i ich rodzaj powinien uwzględniać możliwości obliczeniowe i pamięciowe urządzenia realizującego proces rozpoznawania mowy.

[2] Rozdział 2, zawierający opis teoretyczny technik związanych z rozpoznawaniem mowy, powstał na podstawie pracy magisterskiej autora [105]

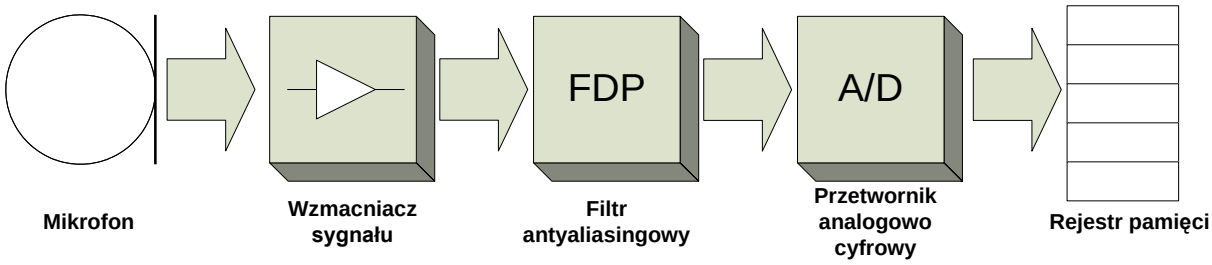
Ostatnim z etapów jest proces **rozpoznawania**. Wybrane parametry sygnałów porównywane są z tymi, które wcześniej zapisano w pamięci. Proces ten może w sobie zawierać także etap uczenia. Na podstawie algorytmów rozpoznawania podejmowana jest decyzja o klasyfikacji sygnału do określonej klasy.

2.1 Etap przetwarzania wstępnego sygnału mowy

Zanim z sygnału zostaną wydobyte najbardziej reprezentatywne parametry, należy wcześniej poddać go procesowi **przetwarzania wstępnego** [14]. Od tego etapu w dużej mierze zależy skuteczność działania całego systemu rozpoznawania mowy.

2.1.1 Akwizycja sygnału mowy

Proces rejestracji sygnału mowy jest pierwszym etapem w procesie jego rozpoznawania. Wejściowy analogowy sygnał mowy, rejestrowany za pomocą mikrofonu, należy przekształcić na jego postać cyfrową. Możliwe jest to dzięki przetwornikowi analogowo/cyfrowemu (A/D). Zanim analogowy sygnał mowy zostanie zapisany w postaci cyfrowej, poddawany jest wcześniej wzmacnianiu, a następnie filtracji filtrem antyaliasingowym (osłonowym). Wszystkie trzy układy realizujące powyższe procesy, tzn. **wzmacniacz wstępny**, **filtr antyaliasingowy** oraz **przetwornik analogowo – cyfrowy**, znajdują się na wyposażeniu standardowej komputerowej karty dźwiękowej. Schemat blokowy akwizycji sygnału przedstawiono na Rys. 2.2.



Rys. 2.2 Schemat blokowy akwizycji sygnału mowy z wykorzystaniem karty dźwiękowej podłączonej do komputera

W czasie konwersji z postaci analogowej na cyfrową sygnał mowy poddawany jest procesowi kwantyzacji i próbkowania.

Próbkowanie (dyskretyzacja czasowa)

Podstawowym parametrem opisującym proces próbkowania sygnału, zwany również dyskretyzacją czasową lub kwantowaniem w czasie, jest **częstotliwość próbkowania** F_p. W latach 90-tych najczęściej stosowanymi w rozpoznawaniu mowy częstotliwościami były 6.67kHz, 8kHz oraz 10kHz [14]. Dziś, dzięki szybkiemu rozwojowi procesorów i pojemności pamięci podręcznej RAM, coraz częściej stosuje się wyższe częstotliwości próbkowania np. 12kHz czy 16kHz. Podczas wyboru częstotliwości próbkowania należy bezwzględnie przestrzegać **twierdzenia o próbkowaniu Kotielnikowa–Shannona,** znanego także jako tw. Whittakera-Nyquista-Kotelnikowa-Shannona. Mówi ono, że częstotliwość próbkowania musi być dwukrotnie większa, bądź równa maksymalnej częstotliwości sygnału

$$F_p \geq 2F_s, \tag{2.1}$$

gdzie:

F_p – częstotliwość próbkowania,

F_s – maksymalna częstotliwość sygnału.

Kwantyzacja (dyskretyzacja amplitudowa)

Kolejnym, równie ważnym co częstotliwość próbkowania parametrem, jest **liczba bitów**. Określa ona, z jaką dokładnością zostanie zapisana wartość próbki. Najczęściej stosowana ilość bitów to 8, 12 lub 16. Projektując system rozpoznawania mowy należy znaleźć kompromis między wiernością zapisu sygnału, a ilością danych zajmujących pamięć i wpływających na szybkość obliczeń.

Kolejnym parametrem akwizycji sygnału jest **ilość kanałów** używanych do nagrywania. Standardowo do procesu rejestracji mowy używa się jednego kanału (sygnał mono) lub dwóch kanałów (sygnał stereo).

Ostatnim z parametrów związanych z akwizycją sygnału jest **czas nagrywania**. Powinien być on na tyle duży, aby użytkownik zdążył nagrać całe słowo. Z drugiej strony czas powinien być ograniczony do minimum, aby nie był rejestrowany dodatkowy szum z otoczenia, oraz aby nie powiększać niepotrzebnie ilości danych do dalszego przetwarzania.

2.1.2 Preemfaza

Po etapie akwizycji zaleca się, aby cyfrowy sygnał mowy poddać filtracji tzw. **filtrem preemfazy**. Proces ten ma na celu ograniczenie szumów tła pochodzących z sieci elektrycznej o częstotliwości 50-60Hz oraz uwydatnienie w widmie składowych wyższych częstotliwości. Dodatkowo dzięki preemfazie eliminowane są niepożądane zjawiska związane z arytmetyką skończonej precyzji.

Skutecznym w działaniu i jednocześnie prostym w realizacji jest cyfrowy **filtr górnoprzepustowy** (ang. *high pass filter*) pierwszego rzędu o skończonej odpowiedzi impulsowej (ang. *Finite Impuls Response – FIR*), którego transmitancja wynosi

$$H(z) = 1 - az^{-1}. \tag{2.2}$$

Równanie różnicowe przyjmuje wtedy postać

$$s[n] = \tilde{s}[n] - a\tilde{s}[n-1], \tag{2.3}$$

gdzie:

$\tilde{s}[n]$ - cyfrowy sygnał mowy przed filtracją dla n – tej próbki,

$s[n]$ - cyfrowy sygnał mowy po filtracji filtrem preemfazy dla n – tej próbki.

Przyjmuje się, że $0.9 < a < 1$. Najczęściej zakłada się współczynnik $a = 0.95$, dzięki czemu możliwe jest wzmocnienie składowych wyższych częstotliwości o ponad 20dB. Czasami stosowane są także inne wartości współczynnika a, np. $a = \frac{15}{16} = 0.9375$ [14].

2.1.3 Wybór początku i końca słowa

Po przejściu sygnału przez filtr preemfazy, sygnał poddawany jest dalszej obróbce. Wybór początku i końca słowa (ang. *end point location*) [14], [15], oznaczany w pracy skrótem VAD (ang. *Voice Activity Detection*), ma na celu ograniczenie rozmiaru sygnału, a dzięki temu zmniejszenie ilości danych, czasu obliczeń i wymaganej pamięci. Co więcej, normalizacja sygnałów co do wartości i czasu ich trwania ma zasadniczy wpływ na jakość rozpoznawania. Słowo wypowiedziane na początku procesu akwizycji sygnału oraz to samo słowo wypowiedziane po dłuższym okresie czasu od rozpoczęcia nagrania będzie dużo trudniej porównać bez bloku redukującego obszar sygnału, w którym brak jest wypowiedzi, a jedynie rejestrowany jest szum.

W ogólności każdy z algorytmów wyboru początku i końca można podzielić na dwa etapy. W pierwszej kolejności obliczane są zadane parametry. W drugim etapie odbywa się klasyfikacja ramek sygnału na podstawie reguły decyzyjnej.

Istnieje wiele metod wyznaczania początku i końca słowa wykorzystujących np.: energię sygnału, liczbę przejść przez zero czy cechy charakterystyczne widma sygnału. Wiele z algorytmów VAD tworzonych jest w oparciu o złożone kryteria decyzyjne wykorzystujące więcej niż jedną z wymienionych metod.

Dobry algorytm wyboru początku i końca wypowiedzi powinien cechować się:

- wysoką niezawodnością,
- prostotą i wysoką szybkością działania,
- możliwością zastosowania w różnych środowiskach akustycznych.

Algorytmy wykorzystujące energię sygnału

Metoda wyboru początku i końca słowa wykorzystująca pomiar energii sygnału jest metodą najbardziej intuicyjną i jednocześnie skuteczną.

Implementacja algorytmów wykorzystujących energię sygnału (ang. *energy – based algorithms*) [16] wymaga wzięcia pod uwagę wielu czynników. Z jednej strony szum ze środowiska różni się od samego sygnału mowy. Z drugiej strony istnieje problem związany ze sposobem wypowiadania przez człowieka słów (ruchy ust).

Cyfrowy sygnał mowy po przejściu przez filtr preemfazy dzielony jest na ramki. Następnie dla każdej z ramek mierzona jest energia ramki (ang. *short-term energy*)

$$E_s[m] = \sum_{n=(m-1)N}^{mN-1} s^2[n], \tag{2.4}$$

gdzie:

N – liczba próbek w ramce,

m – numer ramki,

n – numer próbki sygnału.

Algorytmy wykorzystujące gęstość przejść przez zero

Metoda wykorzystująca **gęstość przejść przez zero** (ang. *Zero Crossing Rate - ZCR*), zwaną niekiedy częstotliwością Rice'a, jest w dużym stopniu podatna na zakłócenia sieci elektrycznej o częstotliwości 50-60Hz. Mimo to metoda ta powinna być w większości przypadków wystarczająca do stwierdzenia, czy jest to sygnał mowy, czy tylko szum tła.

Funkcja przejścia sygnału przez zero $\rho[n]$ wynosi 1, jeżeli między sąsiednimi próbkami n i $n-1$ zachodzi nierówność

$$sgn\{s[n]\} \neq sgn\{s[n-1]\}, \tag{2.5}$$

gdzie sgn oznacza funkcję signum.

Jeżeli warunek (2.5) nie jest spełniony, to $\rho[n] = 0$.

Podobnie jak przy metodzie opartej na pomiarze energii, tu także sygnał dzielony jest na ramki. Następnie liczona jest ilość przejść przez zero w ramce zgodnie z

$$\rho_s[m] = \sum_{n=(m-1)N}^{mN-1} \rho[n]. \tag{2.6}$$

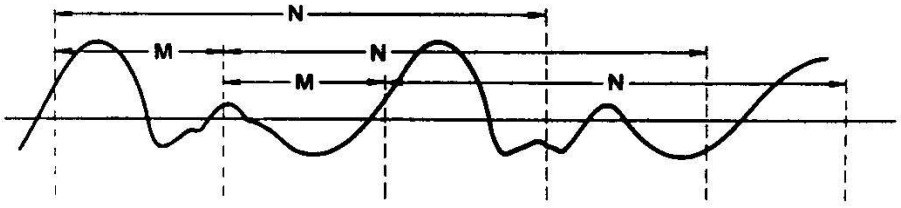
Algorytmy wykorzystujące cechy charakterystyczne widma sygnału

Algorytmy bazujące na cechach charakterystycznych widma sygnału (ang. *Algorithms using frequency – based features*) dają dobre rezultaty przy dużym stosunku sygnału do szumu (ang. *Signal to Nosie Ratio - SNR*). Przy większym zaszumieniu zachowują się jednak gorzej, w porównaniu z algorytmami bazującymi na pomiarze energii sygnału.

2.1.4 Segmentacja

Proces segmentacji (ang. *frame blocking*) sygnału $s[n]$ polega na jego podziale na odpowiedniej długości segmenty (ramki). Ma to na celu uzyskanie sygnału jak najbardziej zbliżonego do stacjonarnego (czyli kwazistacjonarnego) w danym segmencie. Jest to jedno z podstawowych założeń techniki LPC (patrz podrozdz. 2.2.5). Ważne jest przy tym, aby ramki zachodziły na siebie jak pokazano na Rys. 2.3. Najczęściej długość ramki N powiązana jest z jej przesunięciem (skokiem) M zgodnie z zależnością [14 pp. 113 – 114]

$$M = \frac{1}{3}N. \tag{2.7}$$



Rys. 2.3 Sygnał mowy podzielony na ramki o długości *N* i przesunięciu *M*

Typowe wartości parametrów N i M dla kilku częstotliwości próbkowania Fp przedstawia Tab. 2.1 [14 p. 122].

Tab. 2.1 Typowe wartości parametrów używane do segmentacji sygnału

Fp / **Parametr**	**6,67kHz**	**8kHz**	**10kHz**
N	300 (45ms)	240 (30ms)	300 (30ms)
M	100 (15ms)	80 (10ms)	100 (10ms)

Zbyt krótka ramka, o długości poniżej 10 milisekund, będzie powodować zwiększenie ilości obliczeń, powodując dodatkowe obciążenie pamięci i procesora.

Ramka zbyt długa (powyżej 50 milisekund) spowoduje, iż sygnał zawarty w ramce trudno będzie traktować jako sygnał kwazistacjonarny, uniemożliwiając zastosowanie np. techniki LPC na etapie parametryzacji.

Oznaczając l-tą ramkę z sygnałem mowy przez $\tilde{x}_l[n]$, oraz przyjmując, że L określa liczbę wszystkich ramek, można zapisać, że

$$\tilde{x}_l[n] = s[Ml + n], \tag{2.8}$$

gdzie: $n = 0, 1, 2, \ldots, N - 1$ oraz $l = 0, 1, 2, \ldots, L - 1$.

2.1.5 Okienkowanie

Podział sygnału na ramki powoduje powstawanie nieciągłości w przetwarzanym sygnale, co wiąże się z powstawaniem składowych o wyższych częstotliwościach w widmie sygnału. Aby tego uniknąć, sygnał w każdej ramce należy poddać procesowi okienkowania (ang. *windowing*). Możliwe jest to dzięki pomnożeniu zawartego w ramce sygnału przez odpowiednią funkcję okna. Celem okienkowania jest więc minimalizacja błędu estymacji funkcji autokorelacji sygnału na krańcach każdej ramki. Dzięki temu następuje wygładzenie nieciągłości i usunięcie z widma fałszywych składowych. Należy mieć jednak na uwadze, że proces ten wprowadza dodatkowe tłumienie okienkowanego sygnału.

Istnieje wiele rodzajów okien, do których należy m.in.:

- **okno Blackmanna**:

$$w[n] = 0.42 - 0.5\cos(\frac{2\pi n}{N-1}) + 0.08\cos(\frac{4n\pi}{N-1}), \tag{2.9}$$

gdzie $0 \le n \le N-1$,

- **okno Dirichleta (prostokątne)**:

$$w[n] = 1, \text{ dla } |n| \le N$$

oraz (2.10)

$$w[n] = 0 \text{ dla } |n| > N,$$

- **okno Bartletta (trójkątne)**:

$$w[n] = \frac{2n}{N-1} \text{ dla } 0 \le n \le \frac{N-1}{2}$$

oraz (2.11)

$$w[n] = 2 - \frac{2n}{N-1} \text{ dla } \frac{N-1}{2} \le n \le N-1,$$

- **Okno Hanna**:

$$w[n] = \frac{1}{2}\left[1 - \cos(\frac{2\pi n}{N-1})\right]^2, \text{ gdzie } 0 \le n \le N-1, \tag{2.12}$$

- **Okno Papoulisa**:

$$w[n] = \frac{1}{\pi}|\sin(\frac{2\pi n}{N-1})| + \frac{2n}{N-1}\sin(\frac{2\pi n}{N-1}), \text{ gdzie } 0 \le n \le N-1, \tag{2.13}$$

- **Okno Lanczosa**:

$$w[n] = \left[\frac{\sin(\frac{2\pi n}{N-1})}{\frac{2\pi n}{N-1}}\right]^L, \text{ gdzie } 0 \le n \le N-1 \text{ oraz } L \text{ – parametr okna,} \tag{2.14}$$

- **Okno pół-sinusoidy**:

$$w[n] = \frac{\pi}{N}, \text{ gdzie } 0 < n < N, \tag{2.15}$$

- **Okno Hanninga**:

$$w[n] = 0.5 - 0.5\cos(\frac{2\pi n}{N-1}) \text{, gdzie } 0 \le n \le N-1, \quad (2.16)$$

- **Okno Hamminga**:

$$w[n] = a - (1-a)\cos(\frac{2\pi n}{N-1}) \text{, gdzie } 0 \le n \le N-1. \quad (2.17)$$

Dobierając parametr a w oknie Hamminga (2.17) można w pewnym stopniu modulować wpływ składnika prostokątnego i funkcji typu cos^2 [17]. Przyjmuje się, że optymalna wartość współczynnika a wynosi $0{,}54$. Tłumienie dalszych listków bocznych dla tego okna wynosi 20dB/dekadę.

Przy wyborze okna na ogół dąży się do tego, aby widmo funkcji okna czasowego miało jak najmniejsze wstęgi boczne oraz, aby było jak najwęższe.

Rezultat okienkowania danej ramki można zapisać, jako:

$$x_l[n] = \tilde{x}_l[n]w[n], \quad \text{dla } 0 \le n \le N-1. \quad (2.18)$$

2.2 Etap parametryzacji sygnału mowy

Parametryzacja sygnału w procesie rozpoznawania mowy jest niezwykle istotna. Od niej w dużej mierze zależy skuteczność rozpoznawania oraz szybkość działania całego systemu rozpoznawania mowy.

Decydując się na dany typ zbioru parametrów, a nieraz nawet kilku ich typów, należy kierować się:

- zdolnością do uwydatniania cech szczególnych sygnału mowy,
- czasem obliczeń, związanym ze stopniem skomplikowania algorytmu,
- oraz wymaganą ilością pamięci.

Metody parametryzacji sygnału mowy w ogólności można podzielić na trzy główne grupy [18] związane z:

- parametrami czasowymi,
- parametrami częstotliwościowymi,
- oraz parametrami wykorzystującymi liniowe kodowanie predykcyjne.

2.2.1 Parametry czasowe

Ze względu na długość badanego sygnału, metody parametryzacji w dziedzinie czasu można podzielić na dwie grupy analizujące makrostrukturę i mikrostrukturę sygnału.

Parametry dotyczące **makrostruktury** sygnału to:

- natężenie energii sygnału w funkcji czasu,
- oraz obwiednia amplitudy sygnału, która zawiera informacje o zmianach energii sygnału mowy w czasie.

Do parametrów związanych z **mikrostrukturą** sygnału zalicza się:

- liczbę przejść przez zero,
- oraz rozkład interwałów czasowych pomiędzy kolejnymi przejściami przez zero.

2.2.2 Parametry częstotliwościowe

Analiza częstotliwościowa sygnału jest bardzo użytecznym narzędziem do parametryzacji mowy. Wynika to z faktu, iż podczas wytwarzania sygnału mowy zmiany funkcji pobudzenia, a także kształtu kanału głosowego, w znacznym stopniu wpływają na parametry częstotliwościowe sygnału mowy.

Do parametrów częstotliwościowych sygnału zaliczyć można m.in.:

- parametry dyskretnej funkcji autokorelacji,
- parametry cepstrum,
- częstotliwość podstawową tonu krtaniowego,
- częstotliwości formantowe.

2.2.3 Cepstrum

Jedną ze szczególnych metod parametryzacji w dziedzinie częstotliwości jest analiza cepstralna [18], [14], [19]. Jej początki związane są z badaniami Oppenheima nad technikami homeomorficznymi. Ich celem było określenie przekształceń sygnałów w takie przestrzenie wektorowe, w których określone działanie byłoby równoważne dodawaniu.

Dzięki temu do przetwarzania sygnałów można wykorzystać operujące w tych przestrzeniach systemy liniowe, które są ogólnie znane. Można przyjąć, że cepstrum bazuje na uogólnionej zasadzie superpozycji definiowanej dla systemów liniowych. Umożliwia w prosty sposób rozdział składowych addytywnych z użyciem filtracji liniowej.

Początkowo cepstrum wykorzystywano do badania ech sejsmicznych związanych z trzęsieniami ziemi, wybuchami bomb, a także do analizy sygnałów radarowych. Obecnie oprócz analizy sygnałów akustycznych wykorzystywane jest także jako metoda statystyczna do badania okresowości szeregów czasowych.

Cepstrum można podzielić na **cepstrum zespolone** oraz **rzeczywiste**. To ostatnie nazywane jest także **cepstrum mocy**.

Cepstrum zespolone

Cepstrum zespolone można zdefiniować jako

$$C[n] = \left| F^{-1} \ \log(F\{x[n]\}) \ \right|, \tag{2.19}$$

gdzie:

$x[n]$ - dyskretny, poddany procesowi segmentacji i okienkowania sygnał mowy,

F - dyskretna transformata Fouriera,

F^{-1} - odwrotna, dyskretna transformata Fouriera.

W definicji cepstrum zespolonego wymagana jest wartość logarytmu sygnału zespolonego. Dzięki jego znajomości możliwa jest całkowita rekonstrukcja sygnału pierwotnego.

Cepstrum rzeczywiste

Technikę cepstralną, zwaną cepstrum rzeczywistym, opracowali Bogert, Healy i Tukey [20] w 1963 r. Oni też stworzyli terminologię związaną z tą rodziną technik homeomorficznych.

Tab. 2.2.2 Terminologia związana z widmem i cepstrum

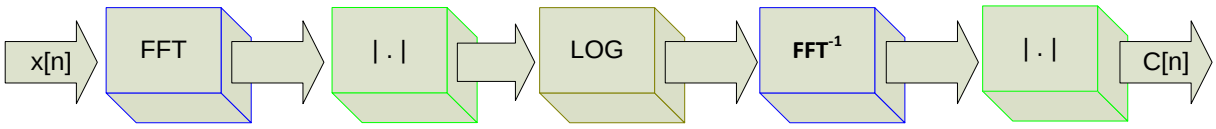
Klasyczna terminologia widmowa	Terminologia związana z cepstrum
SPECTRUM	**CEPS**TRUM
FREQUENCY	**QUE**FRENCY
FILTERING	**LIF**TERING
HARMONICS	**RAH**MONICS

Jak widać w Tab. 2.2.2, nowe słowa powstały z odwrócenia kolejności liter w pierwszej części wybranych słów związanych z dziedziną częstotliwości.

Cepstrum rzeczywiste można zdefiniować jako

$$C[n] = \left| F^{-1} \ \log \left| F\{x[n]\} \right| \ \right|. \tag{2.20}$$

W definicji cepstrum rzeczywistego (2.20) stosowany jest logarytm modułu widma. Ze względu na brak informacji o fazie widma, niemożliwa jest jednak rekonstrukcja pierwotnego sygnału w oparciu o ten typ cepstrum.



Rys. 2.4 Schemat blokowy wyznaczania cepstrum rzeczywistego sygnału

Na Rys. 2.4 przedstawiono schemat blokowy algorytmu, zgodnie z którym obliczane jest cepstrum rzeczywiste sygnału.

Pojęcie cepstrum jest szczególnie przydatne do analizy i syntezy mowy. Przyczyną tego są następujące właściwości cepstrum zespolonego [17 strony 181-182]:

- składowe cepstrum pochodzące od sygnału odpowiedzi impulsowej traktu głosowego skupione są wokół wartości $n = 0$,

- ciąg dyskretnych impulsów równoodległych w czasie ma cepstrum zespolone o tej samej postaci niezależnie od rozkładu amplitud tych impulsów – zmianie ulega jedynie ich amplituda, natomiast impulsy są w tych samych odstępach,
- dla sygnałów spełniających warunek minimalnofazowości można utożsamiać cepstrum rzeczywiste z cepstrum zespolonym.

Wynika z tego m.in., że cepstrum zespolone pobudzenia zawiera impulsy występujące w odstępach odpowiadających okresowi tonu krtaniowego. Cepstrum zespolone odpowiedzi impulsowej kanału głosowego jest skupione wokół $n = 0$, natomiast cepstra zespolone tej odpowiedzi oraz pobudzenia w segmentach dźwięcznych zajmują różne obszary w czasie. Tak więc wartości cepstrum, które reprezentują kanał głosowy, można wydzielić z całkowitego cepstrum za pomocą układu liniowego mnożącego wielkość w czasie wokół $n = 0$ przez jedność, a pozostałe wartości – przez 0.

Parametry pobudzenia, np. tonu krtaniowego, wydziela się z części cepstrum w zakresie dużych wartości n. W segmentach dźwięcznych w cepstrum pojawiają się maksima dla wielokrotności okresu tonu krtaniowego. W segmentach bezdźwięcznych maksima takie nie występują w cepstrum. Sprawdza się więc, czy istnieją maksima cepstrum dla dużych wartości n mając na celu stwierdzenie obecności lub też braku cechy dźwięczności w sygnale mowy. W przypadku wykrycia segmentów dźwięcznych pomiar położenia maksimum cepstrum umożliwia wyznaczenie okresu tonu krtaniowego.

2.2.4 Perceptualne parametry cepstralne

Zgodnie z tzw. **podejściem perceptualnym** uważa się, że najbardziej efektywne wyniki rozpoznawania uzyskuje się naśladując mechanizmy rozpoznawania głosu, które posiada człowiek. Jedną z tego typu metod jest przekształcenie skali częstotliwości w taki sposób, aby odpowiadała ona subiektywnemu odbiorowi częstotliwościowi ludzkiego narządu słuchu.

Stosowane w rozpoznawaniu mowy skale perceptualne charakteryzują się w przybliżeniu liniowym odwzorowaniem częstotliwości niskich oraz logarytmicznym częstotliwości wysokich.

Przekształcenia częstotliwości dokonuje się najczęściej poprzez zastosowanie banku filtrów o częstotliwościach środkowych rozmieszczonych równomiernie na skali perceptualnej (ale nieliniowo na skali częstotliwości), o częściowo nakładających się pasmach przepuszczania.

Cepstrum melowe

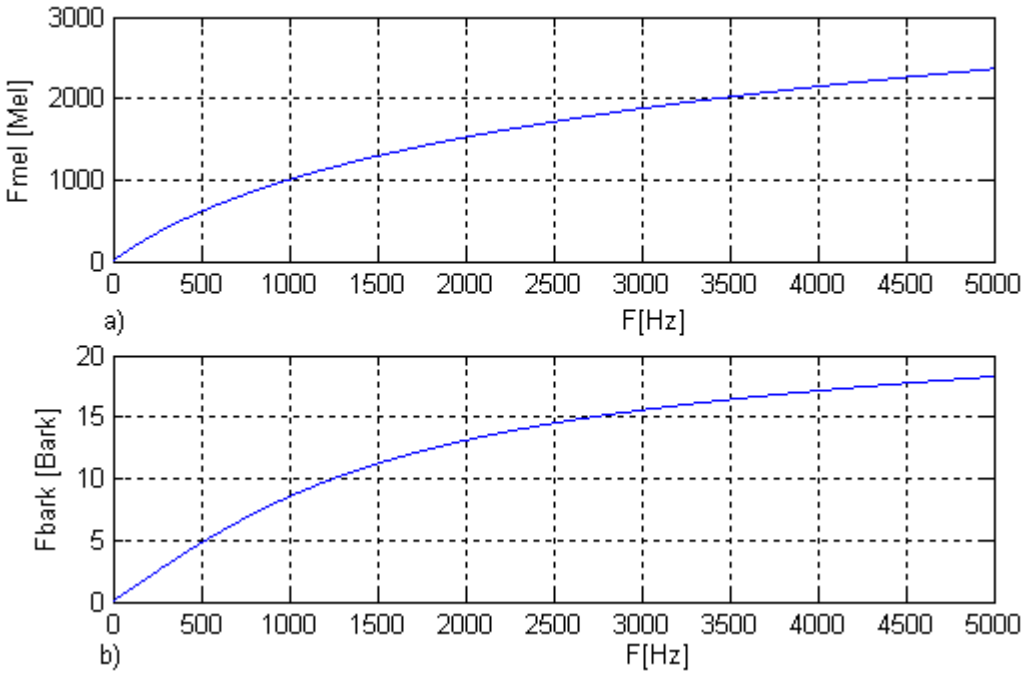
Możliwość podniesienia skuteczności rozpoznawania z użyciem parametryzacji cepstralnej z wykorzystaniem podejścia perceptualnego umożliwia tzw. **cepstrum melowe** [21]. Polega ono na użyciu melowej skali częstotliwości dla widma sygnału. Dopiero po przejściu na skalę melową obliczane jest cepstrum. Zastosowanie tej skali powoduje uwypuklenie składowych niższych częstotliwości w stosunku do wyższych częstotliwości w analizowanym sygnale. Związane jest to z nieliniową percepcją ucha ludzkiego na słyszane częstotliwości różnych sygnałów. Zastosowanie parametrów cepstrum przynosi dobre rezultaty zarówno w systemach rozpoznawania mówców [22] jak i w systemach rozpoznawania mowy [23].

Związek między klasyczną skalą częstotliwości a skalą melową opisuje zależność

$$\boldsymbol{F}_{mel} = 2595 \log_{10}(1 + \frac{\boldsymbol{F}_{Hz}}{700}) \tag{2.21}$$

lub równoważnie

$$\boldsymbol{F}_{mel} = 1127,01048 \ln(1 + \frac{\boldsymbol{F}_{Hz}}{700}) . \tag{2.22}$$



Rys. 2.5 Wykresy ilustrujące związek między skalą melową (rys. a) i barkową (rys. b) a klasyczną skalą częstotliwości

Na Rys. 2.5a) widać, że skala melowa jest liniowa w przybliżeniu do 1000Hz oraz logarytmiczna powyżej. Zgodnie z (2.21) lub (2.22), 1000 meli odpowiada częstotliwości 1000Hz.

W ogólności procedura obliczania współczynników cepstrum melowego MFCC (ang. *Mel-Frequency Cepstral Coefficients*) jest podobna do tej przedstawionej na Rys. 2.4 z uwzględnieniem dodatkowo zamiany przekształconego sygnału na skalę melową. Algorytm z uwzględnieniem tego etapu można opisać następująco:

1) okienkowanie sygnału;
2) obliczenie transformaty Fouriera;
3) obliczenie modułu sygnału;
4) logarytmowanie;
5) przejście na skalę melową;
6) obliczenie odwrotnej transformaty Fouriera;
7) opcjonalne stosowany jest proces zwany lifteringiem.

Oprócz skali *melowej* stosowana jest także czasem skala typu *bark*, którą przedstawiono na Rys. 2.5b).

Definiowana jest ona zgodnie ze wzorem

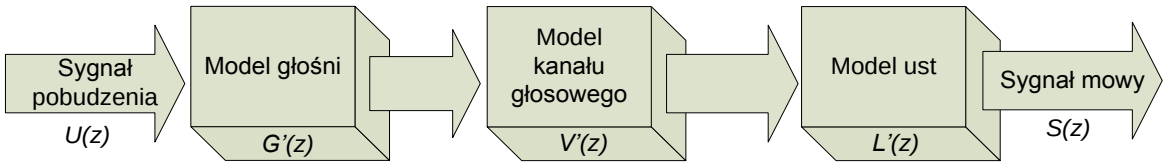
$$\boldsymbol{F}_{\boldsymbol{bark}} = 13\arctan(\frac{0,76\boldsymbol{F}_{\boldsymbol{Hz}}}{1000}) + 3,5\arctan(\frac{\boldsymbol{F}_{\boldsymbol{Hz}}}{7500})^2. \tag{2.23}$$

Inne perceptualne metody parametryzacji

Oprócz wymienionych technik parametryzacji istnieje także wiele innych [24] nieopisanych w niniejszej pracy. Przykładem tego typu technik może być np. metoda parametryzacji zwana PLP (ang. *Perceptual Linear Prediction*), zaproponowana przez Hynka Hermansky'iego [25]. Kolejną techniką zaproponowaną przez niego wraz Morganem jest tzw. RASTA (ang. RelAtive SpecTrA) [26]. Dzięki zastosowaniu cyfrowego filtru pasmowo-przepustowego technika ta uwydatnia szczególnie zmiany cepstrum o częstotliwościach rzędu 2-6Hz. Częstotliwość 4-6Hz jest typowa w trakcie wypowiadania kolejnych sylab. W wyniku połączenia ze sobą wybranych elementów obu metod powstała nowa technika nazywana RASTA-PLP [27], [28]. Ze względu na fakt, że metody te nie będą testowane w niniejszej pracy, pominięto ich szerszy opis.

2.2.5 Liniowe Kodowanie Predykcyjne

Liniowe kodowanie predykcyjne (ang. *Linear Predictive Coding – LPC*), nazywane też **liniową predykcją** [14], [15], [19], [29], [30], zastosowali po raz pierwszy Saito i Itakura w 1966 roku oraz Atal w 1967 r. Z początkiem lat siedemdziesiątych LPC stała się powszechnie używanym narzędziem w przetwarzaniu, analizie i syntezie sygnału mowy. Metoda LPC zawdzięcza swoją popularność m.in. możliwości modelowania za pomocą wielobiegunowej struktury funkcji transmitancji kanału głosowego. Model ten, bazujący na analizie budowy narządu mowy człowieka, opisany został przez Fanta i Flanagana. Uproszczony schemat blokowy modelu generacji mowy przedstawiono na Rys. 2.6.



Rys. 2.6 Model generacji mowy

W dużym uproszczeniu można założyć, że

$$S(z) = [G'(z)V'(z)L'(z)]U(z). \tag{2.24}$$

W czasie prac nad modelem generacji mowy przyjęto pewne uproszczenia. Założono, że kształt kanału głosowego jest stały, a sam kanał można uważać za liniowy. W takim przypadku odpowiedź (reakcja) kanału na pobudzenie jest splotem pobudzenia i odpowiedzi impulsowej kanału głosowego. Kolejnym uproszczeniem jest założenie, że sygnał w analizowanym przedziale czasu jest stacjonarny. Aby zrealizować to założenie w praktyce, sygnał dzielony jest na ramki. Przyjmuje się także, że transmitancja głośni $G'(z)$, a także transmitancja ust $L'(z)$ są stałe. Wynika stąd, iż sygnał

mowy zależny jest tylko od transmitancji kanału głosowego oraz pobudzenia. Najważniejszym z założeń jest możliwość wyznaczenia charakterystyki kanału głosowego na podstawie informacji o kilku pierwszych formantach.

Model Fanta – Flanagana jest modelem o samych biegunach. Odwrotność jego funkcji transmitancji zapisać można w postaci wielomianu ujemnych potęg z zgodnie z wzorem:

$$A(z) = \sum_{k=0}^{M} \alpha_k z^{-1} \quad , \tag{2.25}$$

gdzie $\alpha_0 = 1$ oraz

$$M = 2K + 1. \tag{2.26}$$

M oznacza stopień wielomianu, a K jest liczbą formantów uwzględnionych w modelu. Wielomian $A(z)$ jest transmitancją **filtru inwersyjnego** (odwrotnego). Wynika to z odwrotnej zależności wielomianu $A(z)$ od funkcji transmitancji modelu generacji mowy. Współczynniki wielomianu $\{\alpha_k\}$, razem z parametrami opisującymi rodzaj pobudzenia $U(z)$ a także wzmocnieniem G, określają biegunowy model syntezy mowy, który można opisać zależnością pomiędzy transformatą pobudzenia $U(z)$ i transformatą reakcji $S(z)$ zgodnie z

$$S(z) = \frac{G}{A(z)} U(z). \tag{2.27}$$

Model Fanta-Flanagana nie jest idealny. Nie w pełni odzwierciedla on dźwięki nosowe, dla których powinno się obok biegunów uwzględnić także zera transmitancji toru głosowego. W większości przypadków może on być jednak stosowany z dużą skutecznością do większości dźwięków mowy.

Zaletą stosowaniu modelu $S(z)$ (2.27) jest możliwość szybkiego i skutecznego wyznaczenia wzmocnienia G, jak i współczynników $\{\alpha_k\}$ wielomianu z użyciem urządzeń wyposażonych w procesory (np. komputery czy telefony komórkowe).

Z biegunowego modelu syntezy mowy wynika, że sygnał $s[n]$ może być przedstawiony w formie równania różnicowego

$$s[n] = -\sum_{k=1}^{M} \alpha_k s[n-k] + Gu[n]. \tag{2.28}$$

Jednym z założeń w LPC jest to, że próbka sygnału może być w przybliżeniu przedstawiona jako liniowa kombinacja poprzednich próbek, co można zapisać jako

$$\hat{s}[n] = \sum_{k=1}^{M} a_k s[n-k]. \tag{2.29}$$

Różnica między sygnałem oryginalnym $s[n]$ a estymowanym $\hat{s}[n]$ nazywana jest błędem predykcji $e[n]$, który można zapisać w postaci

$$e[n] = s[n] - \hat{s}[n] = s[n] - \sum_{k=1}^{M} a_k s[n-k]. \tag{2.30}$$

Błąd $e[n]$ można traktować jako sygnał wyjściowy filtru o transmitancji

$$A(z) = 1 - \sum_{k=1}^{M} a_k z^{-k}. \tag{2.31}$$

Można zauważyć, że jeżeli sygnał $s[n]$ jest taki sam jak sygnał modelowany przez równanie (2.28) i jeżeli $\alpha_k = -a_k$, to $e[n] = Gu[n]$. Jednocześnie filtr $A(z)$ jest odwrotnością w stosunku do transmitancji $H(z)$ modelu generacji mowy

$$H(z) = \frac{S(z)}{U(z)} = \frac{G}{A(z)}. \tag{2.32}$$

Podstawowy problem analizy LPC polega na takim doborze współczynników {α_k}, aby jak najlepiej minimalizować średniokwadratowy błąd predykcji definiowany jako

$$E = \sum_n e^2[n] = \sum_n \left(s[n] - \sum_{k=1}^{M} a_k s[n-k] \right)^2. \tag{2.33}$$

W tym celu kolejne pochodne cząstkowe względem a_k przyrównywane są do zera

$$\frac{\partial E}{\partial a_k} = 0, \tag{2.34}$$

dla $k = 1, \ldots, M$.

Końcowym efektem powyższego procesu jest układ nazywany **układem równań normalnych Youle'a – Walkera** o postaci

$$\sum_{k=1}^{M} a_k \Phi[j,k] = \Phi[j,0], \text{ dla } j = 1, \ldots, M, \tag{2.35}$$

gdzie:

$$\Phi[j,k] = \sum_n s[n-j] s[n-k]. \tag{2.36}$$

Układ równań Youle'a–Walkera opisuje układ M równań liniowych z M niewiadomymi. Rozwiązaniem układu jest M współczynników predykcji.

Znanych jest kilka metod obliczania współczynników predykcji, z których powszechnie stosuje się dwie: metodę kowariancyjną oraz metodę korelacyjną.

Główną różnicą między tymi dwoma metodami jest zastosowanie okna czasowego dla metody korelacyjnej. Metoda kowariancyjna nie wymaga okienkowania.

Metoda kowariancyjna

Zaletą metody kowariancyjnej jest możliwość dokładnego odtworzenia ciągu sygnału, który może być modelowany jako odpowiedź impulsowa systemu o transmitancji zawierającej same bieguny. Warunek ten nie jest w pełni możliwy do realizacji w metodzie korelacyjnej.

Wadą metody kowariancyjnej jest to, iż macierz współczynników nie jest macierzą Toeplitza. Z tego powodu w praktyce częściej do wyznaczania współczynników predykcji korzysta się z metody korelacyjnej.

Metoda korelacyjna

W metodzie korelacyjnej konieczne jest poddanie sygnału wcześniejszemu okienkowaniu. Okno czasowe powinno być na tyle krótkie, by można było założyć, iż sygnał w obrębie okna jest lokalnie stacjonarny. W przypadku sygnału mowy dla segmentów dźwięcznych (pobudzenie krtaniowe) długość okna powinna być równa okresowi pobudzenia (lub też kilku jego kolejnym okresom). W przeciwnym wypadku występuje zjawisko uzależnienia wektora parametrów od położenia impulsów krtaniowych, tzw. efekt zdudnienia.

W metodzie korelacyjnej zakłada się, iż próbki sygnału przyjmują wartości niezerowe w okienkowanym przedziale $0 \leq n \leq N-1$, a poza nim wartości próbek przyjmują wartość zero. W związku z tym może powstać dość znaczny błąd. Minimalizacji tego błędu dokonuje się dzięki zastosowaniu okna czasowego, które gładko sprowadza sygnał do zera na swoich końcach.

Istnieje kilka zalet stosowania metody autokorelacyjnej. Po pierwsze, współczynniki autokorelacji są elementami macierzy Toeplitza. Oznacza to, iż elementy wzdłuż każdej przekątnej macierzy są jednakowe. Zatem przy obliczaniu elementów macierzy wystarczy wyznaczyć tylko jeden wiersz. Kolejną zaletą jest istnienie skutecznych algorytmów iteracyjnych wyznaczania współczynników predykcji. Metoda zapewnia także teoretyczną stabilność systemu o samych biegunach (zakładając nieskończoną dokładność obliczeń).

Klasycznym rozwiązaniem liniowego układu M równań z M niewiadomymi jest zastosowanie **metody eliminacji Gaussa**. Wadą tego rozwiązania jest duża ilość obliczeń proporcjonalna do $\frac{M^3}{3}$. To, iż w metodzie autokorelacji macierz Φ jest macierzą Toeplitza, znacznie upraszcza proces i redukuje czas obliczeń.

Istnieje kilka metod obliczania współczynników predykcji w metodzie korelacyjnej. W ogólności można je podzielić na dwa typy:

- **metody dwuetapowe** – np. algorytmy Durbina (Levinsona i Robinsona),
- oraz **metody kratowe** (ang. *lattice methods*) – np. metoda częściowych współczynników (ang. *PARtial CORrelation coefficients – PARCOR*).

Warto wspomnieć, że nie są to jedyne metody. Autor w [31] przedstawił wyniki własnych badań związanych z zastosowaniem niestacjonarnych metod iteracyjnych, takich jak np. metody gradientu sprzężonego, do wyznaczania współczynników predykcji na podstawie korelacji sygnału.

Rekursywna metoda Durbina

Rekursywna metoda Durbina wykorzystuje własności macierzy korelacyjnej, będącej macierzą Toeplitza. Algorytm obliczania współczynników predykcji w oparciu o tę metodę wygląda następująco.

Warunek początkowy:

$$E^{(0)} = R[0] \qquad \text{- błąd predykcji} \tag{2.37}$$

Procedura rekurencyjna:

$$k_i = \frac{R[i] - \sum_{j=1}^{i-1} a_j^{(i-1)} R[i-j]}{E^{(i-1)}}, \tag{2.38}$$

$$a_i^{(i)} = k_i, \tag{2.39}$$

$$a_j^{(i)} = a_j^{(i-1)} - k_i a_{i-j}^{(i-1)}, \tag{2.40}$$

dla $1 \le j \le i-1$

$$E^{(i)} = (1-k_i)^2 E^{(i-1)}, \tag{2.41}$$

dla $1 \le i \le M$, gdzie:

k_i – współczynniki odbicia między kolejnymi członami tuby akustycznej,

a_i – współczynniki predykcji.

Rozwiązaniem jest zbiór współczynników $a_j = a_j^{(M)}$, dla $1 \le i \le M$. Jak widać, dodatkowo wyznaczany jest błąd predykcji $E^{(i)}$ w i-tej iteracji.

2.2.6 Liniowe kodowanie predykcyjne cepstrum

Możliwe jest obliczenie współczynników cepstralnych na podstawie znajomości parametrów liniowego kodowania predykcyjnego w sposób bezpośredni [32] jak i rekurencyjny [14], [33]. Ten ostatni zostanie pokrótce przedstawiony. Współczynniki cepstralne, uzyskane na podstawie współczynników filtru predykcyjnego, będą dalej nazywane współczynnikami liniowego kodowania predykcyjnego cepstrum, w skrócie LPCC (ang. *Linear Predictive Cepstrum Coding*) i oznaczane jako c_n.

Zdefiniowano transmitancję filtru inwersyjnego $A(z)$ zgodnie z (2.25) przy założeniach, że $\alpha_0 = 1, \alpha_M \neq 0$. Wtedy

$$H(z) = \frac{1}{A(z)} \tag{2.42}$$

jest transmitancją filtru o samych biegunach, którego współczynniki cepstrum c_n są zdefiniowane poprzez zależność

$$\ln[H(z)] = \sum_{n=1}^{\infty} c_n z^{-n} . \tag{2.43}$$

Poprzez różniczkowanie (2.43) można otrzymać zależność rekurencyjną określającą kolejne współczynniki LPCC

$$c_n = -a_n - \frac{1}{n}\sum_{k=1}^{n-1} k c_k a_{n-k} \ , \text{ dla } n > 0, \tag{2.44}$$

gdzie: $a_0 = 1$ oraz $a_k = 0\ dla\ k > M$.

Współczynniki LPCC z powodzeniem stosowane są do zagadnień związanych z rozpoznawaniem mowy. Według [34] osiągają lepsze wyniki jako parametry niż tradycyjne współczynniki filtru predykcyjnego a_k oraz współczynniki odbicia k_i.

2.2.7 Alternatywne metody reprezentacji współczynników predykcji

W technice telekomunikacyjnej, współczynniki LPC rzadko używane są w sposób bezpośredni. Zamiast tego często stosuje się inne parametry, które mogą zostać obliczone na podstawie współczynników filtru predykcyjnego a_k.

Bieguny funkcji transmitancji *H(z)*

Transmitancję filtru inwersyjnego $A(z)$ można przedstawić w postaci iloczynowej

$$A(z) = \sum_{i=1}^{M} a_k z^{-i} = \prod_{i=1}^{M} (1 - z_i z^{-1}) . \tag{2.45}$$

Zera takiego wielomianu z_i mogą przyjmować wartości rzeczywiste, bądź mogą być parami zespolonych wartości sprzężonych.

Mając dane zera wielomianu z_i, można obliczyć częstotliwości formantowe zgodnie z zależnością

$$F_i = \frac{1}{2\pi T} \mathrm{Im}[\ln z_i] , \tag{2.46}$$

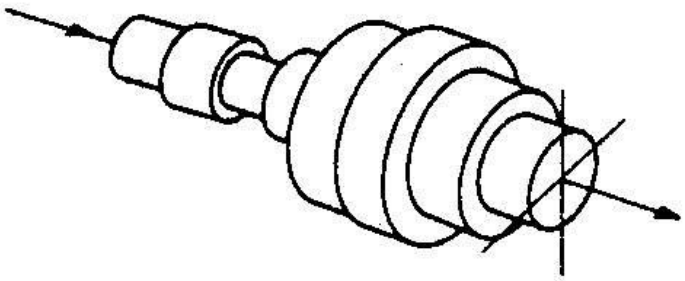
oraz ich pasma częstotliwości

$$B_i = \frac{1}{2\pi T} \mathrm{Re}[\ln z_i] , \tag{2.47}$$

gdzie T – okres próbkowania.

Współczynniki odbicia

Współczynniki odbicia (ang. *Reflection Coefficients - RC*) [14], [15], [30] charakteryzują zmiany przekroju tuby akustycznej.



Rys. 2.7 Grupa cylindrycznych sekcji interpretowana jako model tuby akustycznej

Odpowiadają one współczynnikom k_i (2.38) powstałym podczas rekursywnej procedury Durbina lub w metodzie PARCOR.

W metodzie częściowych współczynników korelacji (PARCOR) zakłada się, że każdą próbkę sygnału można przewidzieć na podstawie danej liczby próbek ją poprzedzających lub następujących po niej. Z tego powodu definiowane są dwa rodzaje błędów predykcji, nazywane błędami *„w przód*" i w *„wstecz*". Wartość błędu w danym kroku zależy od wartości błędu go poprzedzającego. Zaletą tej metody jest możliwość wyznaczenia współczynników odbicia k_i bez potrzeby wyznaczania współczynników predykcji.

Kolejną zaletą jest, że wszystkie parametry k_i charakteryzują się numeryczną stabilnością, tzn.

$$\left|k_i\right| \leq 1 \text{ dla } i = 1,2,\ldots,M. \tag{2.48}$$

Parametry odwrotnej funkcji sinus

Parametry odwrotnej funkcji sinus σ_i (ang. *Inverse Sine Parameters - ISP*) [15] obliczane są na podstawie współczynników odbicia k_i i vice versa, tzn. współczynniki k_i można obliczyć na podstawie ISP. Parametry σ_i definiuje się jako:

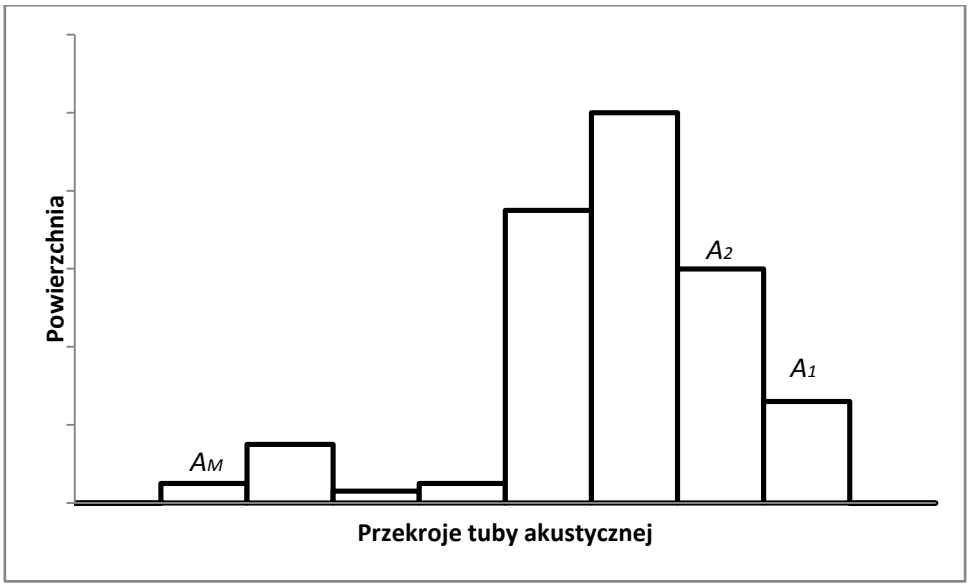
$$\sigma_i = \frac{2}{\pi}\arcsin(k_i) \text{ dla } i = 1,2,\ldots,M. \tag{2.49}$$

Dużą zaletą parametrów ISP jest fakt, że parametry te po przekształceniu pozostają w środku koła jednostkowego.

Zarówno parametry ISP, jak i opisane dalej współczynniki logarytmicznego stosunku powierzchni przekrojów tuby akustycznej (LAR), stosowane są głównie w kodowaniu mowy.

Współczynniki logarytmicznego stosunku powierzchni przekrojów tuby akustycznej

Podobnie jak parametry odwrotnej funkcji sinus σ_i, tak i **parametry logarytmicznego stosunku powierzchni przekrojów tuby akustycznej** (ang. *Log Area Ratio Parameters – LAR*) [14], [15] obliczane są na podstawie współczynników odbicia k_i. Istnieje także możliwość obliczenia współczynników k_i ze współczynników logarytmicznego stosunku powierzchni $\log g_i$.



Rys. 2.7 Funkcja przekroju tuby akustycznej

Jeżeli oznaczy się przekrój i-tej sekcji tuby A_i (Rys. 2.7), to współczynniki odbicia k_i można zdefiniować jako

$$k_i = \frac{A_i - A_{i+1}}{A_i + A_{i+1}}. \tag{2.50}$$

Wzór (2.50) daje możliwość bezpośredniego wyznaczania kolejnych przekrojów tuby zgodnie z zależnością

$$A_{i+1} = \frac{1-k_i}{1+k_i} A_i. \tag{2.51}$$

Na tej podstawie można wyznaczyć stosunek powierzchni dwóch znajdujących się obok siebie przekrojów tuby akustycznej

$$g_i = \frac{A_{i+1}}{A_i} = \frac{1-k_i}{1+k_i} \tag{2.52}$$

dla $i = 1, 2, \ldots, M$, gdzie M – ilość sekcji.

Na podstawie wzoru (2.52) logarytmiczny stosunek powierzchni przekrojów tuby akustycznej można zdefiniować jako

$$\log(g_i) = \log(\frac{A_{i+1}}{A_i}) = \log(\frac{1-k_i}{1+k_i}). \tag{2.53}$$

Zarówno parametry ISP jak i LAR, mimo iż są tylko inną formą współczynników odbicia k_i, bardzo często używane są zamiast tych ostatnich. Dzieje się tak, ponieważ jeżeli współczynnik odbicia jest bliski jedności, to wyniki bardzo podatne są na błąd kwantyzacji. Powyższe przekształcenia „zwijają” amplitudową skalę parametrów, zmniejszając przez to wrażliwość na błąd.

Częstotliwości widma liniowego

W latach 80-tych częstotliwości widma liniowego (ang. *Line Spectral Frequencies – LSF*) [14 p. 191], [15 pp. 331 – 332], [35], [36], [37] (inna nazwa to ang. *Line Spectrum Pair - LSP*) zostały wprowadzone jako alternatywna reprezentacja współczynników predykcji. Metoda ta była intensywnie rozwijana szczególnie przez japoński przemysł telefoniczny.

Współczynniki LSF są zerami dwóch wielomianów utworzonych z użyciem filtru inwersyjnego o transmitancji $A(z)$ zgodnie z (2.31). Definiując je jako:

$$P(z) = A(z) + z^{-(M+1)}A(z^{-1}) \tag{2.54}$$

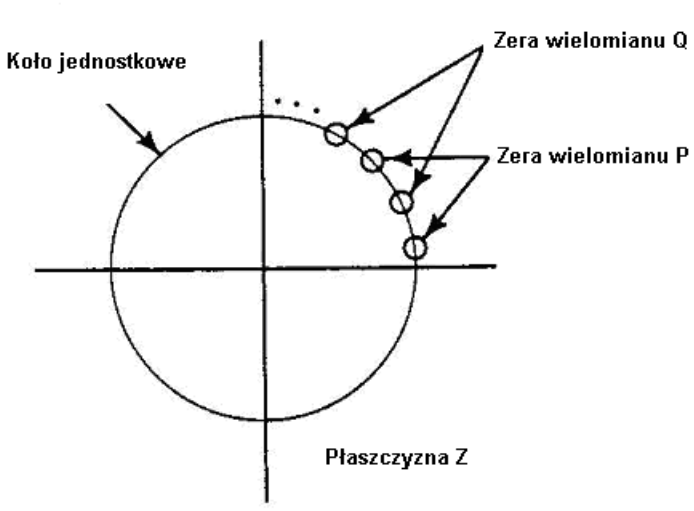
oraz

$$Q(z) = A(z) - z^{-(M+1)}A(z^{-1}) \tag{2.55}$$

można zapisać, że

$$A(z) = \frac{P(z) - Q(z)}{2}. \tag{2.56}$$

Wielomiany $P(z)$ (2.54) i $Q(z)$ (2.55) odpowiadają sztucznemu wydłużeniu tuby o M przekrojach. Dodatkowy przekrój jest albo całkowicie zamknięty, wtedy jego powierzchnia wynosi 0, bądź też całkowicie otwarty. W tym ostatnim przypadku jego powierzchnia jest nieskończona. W związku z tym wszystkie zera wielomianów $P(z)$ i $Q(z)$ leżą na kole jednostkowym na płaszczyźnie z. W rzeczywistości wielomian $P(z)$ ma zero dla *z = -1*, a wielomian $Q(z)$ dla *z = 1*. Pozostałe zera obu wielomianów są zespolone i przeplatają się wzajemnie tak, jak to przedstawiono na Rys. 2.8.



Rys. 2.8 Zera wielomianów *P* i *Q* na płaszczyźnie *Z*

Chociaż zera wielomianu są zespolone, to ich moduły są równe 1 leżąc na kole jednostkowym. Do opisu wielomianów wystarczy więc tylko jeden parametr (częstotliwość lub kąt).

Stwierdzono, iż metoda ta jest o około 30% wydajniejsza w kodowaniu niż w przypadku użycia współczynników przekroju tuby akustycznej [15 p. 332].

Parametry LSF, ze względu na ich specyficzną strukturę, posiadają właściwości zbliżone do częstotliwości formantowych i szerokości pasma.

2.3 Etap klasyfikacji

Klasyfikacja jest ostatnim z etapów w procesie rozpoznawania mowy. To tu parametry charakterystyczne sygnałów są porównywane ze sobą. Na podstawie otrzymanych wyników podejmowana jest decyzja o klasyfikacji do danej klasy zgodnie z przyjętą regułą decyzyjną.

2.3.1 Podział systemów automatycznego rozpoznawania obrazów

Systemy automatycznego rozpoznawania mowy (ang. *Automatic Speech Recognition - ASR*) należą do większej grupy systemów automatycznego rozpoznawania obrazów, które można podzielić na systemy:

- proste, w których parametry obrazu są tego samego rodzaju dla wszystkich klas obiektów rozpoznawalnych, a różnią się tylko wartościami,
- złożone, w których parametry wejściowe są różnego rodzaju, np. zarówno częstotliwościowe, jak i czasowe.

Dodatkowo systemy złożone można podzielić na systemy:

- **jednopoziomowe**, które do rozpoznawania wykorzystują informację zawartą w pierwotnych parametrach pomiarowych (bez dalszego ich przetwarzania),
- **wielopoziomowe (hierarchiczne)**, w których parametry pierwotne pogrupowane w podklasy służą do rozpoznawania na poziomie pierwotnym, a otrzymane wyniki służą jako dane wejściowe do kolejnego poziomu rozpoznawania.

Ze względu na sposób uczenia systemy ASR można podzielić na systemy:

- **z nauczycielem** – wykorzystujące opis klas ustalony a priori,
- **bez nauczyciela** – używane, gdy z założenia nie dysponuje się informacją o przynależności do określanej klasy danych wejściowych.

Ze względu na stosowany algorytm rozpoznawania, systemy rozpoznawania dzielą się na:

- **logiczne** - o logicznych kryteriach podejmowania decyzji,
- **strukturalne (syntaktyczne)**,
- **probabilistyczne** - o statystycznych kryteriach podejmowania decyzji.

Systemy ASR można podzielić także na takie, które rozpoznają **mowę ciągłą** oraz te, które rozpoznają **izolowane słowa**. Do tych ostatnich należy system będący przedmiotem rozważań w niniejszej pracy.

Typowy system ASR jest systemem złożonym lub prostym, jedno- lub wielopoziomowym, uczącym się z nauczycielem, o statystycznym kryterium podejmowania decyzji.

2.3.2 Algorytmy rozpoznawania

Statystyczne algorytmy rozpoznawania można podzielić na parametryczne i nieparametryczne. Dla **algorytmów parametrycznych** gęstości prawdopodobieństw warunkowych $P(x|m)$, gdzie x jest obrazem wypowiedzi w wielowymiarowej przestrzeni, a m - numerem klasy obiektów, są znane lub estymowane na podstawie ciągu uczącego. Dla **algorytmów nieparametrycznych** nie ma potrzeby tworzenia założeń dotyczących typu rozkładów.

Algorytm Bayesa

Jednym z najbardziej znanych algorytmów statystycznych jest **algorytm rozpoznawania Bayesa**. Rozpoznawanie według tego algorytmu polega na przyporządkowaniu losowo pojawiającemu się obrazowi decyzji o jego przynależności do jednej z wcześniej zdefiniowanych klas. Dla zaobserwowanego obrazu estymowane jest prawdopodobieństwo warunkowe przynależności do danej klasy. Obraz kwalifikowany jest do klasy, dla której prawdopodobieństwo jest największe. Wadami algorytmu Bayesa są: złożoność procedury obliczeń oraz potrzeba znajomości rozkładów warunkowych a priori.

Inaczej działają **nieparametryczne algorytmy rozpoznawania**, do których należą m.in.: algorytm *NN, k-NN* czy *NM*.

Algorytm *NN*

W algorytmie „najbliższy sąsiad" (*NN* - ang. *Nearest Neighbour*) w czasie procesu uczenia zapamiętywany jest cały ciąg uczący. Procedura algorytmu oblicza funkcję podobieństwa, którą najczęściej jest odległość pomiędzy wszystkimi obrazami ciągu uczącego a nieznanym obrazem. Po obliczeniu wszystkich funkcji odległości wyszukiwana jest najmniejsza z nich. Klasyfikator podejmuje decyzje o przydziale nazwy, kodu lub numeru klasy, do której należał obraz ciągu uczącego, który okazał się najbardziej bliski w sensie funkcji podobieństwa do obrazu rozpoznawanego. Algorytm *NN* jest jednym z najbardziej wymagających algorytmów stosowanym w rozpoznawaniu mowy, pod kątem rozmiaru pamięci i możliwości obliczeniowych jednostki obliczeniowej [37].

Algorytm *k–NN*

Modyfikacją algorytmu *NN* jest algorytm „*k-najbliższych sąsiadów*" *k-NN*. Pozwala on zmniejszyć wrażliwość systemu rozpoznawania w stosunku do ciągu uczącego. Procedura algorytmu *k-NN* dokonuje obliczeń odległości pomiędzy obrazem rozpoznawanym a wszystkimi obrazami ciągu uczącego i porządkuje te odległości w kolejności rosnącej. Następnie rozpatrywanych jest k pierwszych wartości odległości, dla których określa się, ile z nich odpowiada poszczególnym klasom. Klasyfikator wybiera tę klasę, która najczęściej pojawiała się wśród k pierwszych odległości.

Algorytm *NM*

W algorytmie „najbliższa średnia" (*NM* - ang. *Nearest Mean*) wzorcem klasy rozpoznawanych obiektów jest wartość średnia. Podobnie też jak w algorytmie *NN* obliczane są odległości obrazu rozpoznawanego od wszystkich obrazów wzorcowych (średnich), a następnie wybierana jest najmniejsza z nich.

Podstawowymi zaletami algorytmu *NM* w stosunku do *NN* jest brak konieczności pamiętania wszystkich obrazów ciągu uczącego oraz mniejsza ilość obliczeń.

2.3.3 Funkcja podobieństwa

Funkcja podobieństwa umożliwia realizację **reguły decyzyjnej**. Może przybierać postać funkcji miary odległości lub funkcji bliskości.

Najczęściej stosowaną miarą odległości m.in. w rozpoznawaniu mowy jest odległość **średniokwadratowa** (**Euklidesa**), definiowana jako

$$D(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{p=1}^{P}(x_p - y_p)^2}\,, \tag{2.57}$$

gdzie P jest wymiarem wektorów $\mathbf{x}$ i $\mathbf{y}$. Metoda ta, choć wydaje się najbardziej intuicyjna, posiada pewne wady. Operacja podnoszenia do kwadratu a następnie pierwiastkowania zwiększa dodatkowo czas obliczeń. Aby temu zaradzić, stosuje się niekiedy metrykę uliczną, nazywaną także miarą Hamminga ze względu na wprowadzoną przez tego badacza miarę odległości ciągów kodowych. **Odległość uliczna** (ang. *city block)* definiowana jest jako

$$D(\mathbf{x},\mathbf{y}) = \sum_{p=1}^{P}\left|x_p - y_p\right|. \tag{2.58}$$

Uogólnieniem dwóch wspomnianych metod jest **odległość Minkowskiego**, którą można zdefiniować jako

$$D(\mathbf{x},\mathbf{y}) = \left(\sum_{p=1}^{P}\left|x_p - y_p\right|^{r}\right)^{1/r}, \tag{2.59}$$

dla której dla $r = 2$ otrzyma się odległość średniokwadratową, a dla $r = 1$ miarę uliczną.

Odległość Minkowskiego (więc także miara euklidesowa i uliczna) mają pewną wadę. Jeśli zakres zmienności któregoś z parametrów jest większy niż innego, to odpowiednie składniki w sumie będą dominować nad innymi, co może nie odpowiadać faktycznej ich hierarchii ważności.

Wady tej w pewnym stopniu pozbawiona jest odległość **Camberra,** definiowana jako

$$D(\mathbf{x},\mathbf{y}) = \sum_{p=1}^{P}\frac{\left|x_p - y_p\right|}{\left|x_p + y_p\right|}. \tag{2.60}$$

Charakteryzuje się ona „samonormalizacją" poszczególnych cech, a przy tym jest prosta obliczeniowo.

Specjalnie dla potrzeb liniowego kodowania predykcyjnego Itakura opracował miarę odległości, którą można zdefiniować jako

$$D(\mathbf{x},\mathbf{y}) = \log(\frac{\mathbf{a}_x \mathbf{C}_x \mathbf{a}_x^T}{\mathbf{a}_y \mathbf{C}_y \mathbf{a}_y^T}) , \tag{2.61}$$

gdzie:

$\boldsymbol{C}_x$, $\boldsymbol{C}_y$ – macierze autokorelacji wektorów $\mathbf{x}$ i $\mathbf{y}$,

$\boldsymbol{a}_x$, $\boldsymbol{a}_y$ – wektory współczynników predykcji wektorów $\mathbf{x}$ i $\mathbf{y}$.

Odległość ta, nazwana od nazwiska autora **miarą Itakury**, jest efektywna przy zastosowaniu do parametrów LPC. Posiada jednak poważną wadę. Jej zastosowanie wymaga użycia macierzy korelacji dla wektorów $\mathbf{x}$ i $\mathbf{y}$. W rzeczywistości konieczne byłoby pamiętanie macierzy autokorelacji wszystkich sygnałów ze słownika od momentu analizy LPC aż do zakończenia całego procesu rozpoznawania.

Stosowane są także inne miary odległości. Niektóre z nich to:

- odległość **Mahalanobisa:**

$$D(\mathbf{x},\mathbf{y}) = (\mathbf{x}-\mathbf{y})^T \mathbf{C}^{-1} (\mathbf{x}-\mathbf{y}) , \tag{2.62}$$

gdzie: $\boldsymbol{C}$ – macierz kowariancji wzajemnej wektorów $\mathbf{x}$ i $\mathbf{y}$,

- odległość **Tanimoto:**

$$D(\mathbf{x},\mathbf{y}) = \frac{\mathbf{x}^T\mathbf{y}}{\mathbf{x}^T\mathbf{x} + \mathbf{y}^T\mathbf{y} - \mathbf{x}^T\mathbf{y}} , \tag{2.63}$$

- odległość **Chi-kwadrat:**

$$D(\mathbf{x},\mathbf{y}) = \sum_{p=1}^{P} \frac{1}{x_p - y_p} \left(\frac{x_p}{\sum_{r=1}^{p} x_r} - \frac{y_p}{\sum_{r=1}^{p} y_r} \right) . \tag{2.64}$$

W [38] autor przedstawił wyniki własnych badań nad wpływem doboru miary odległości na wyniki skuteczności rozpoznawania, przy zastosowaniu różnych metod parametryzacji.

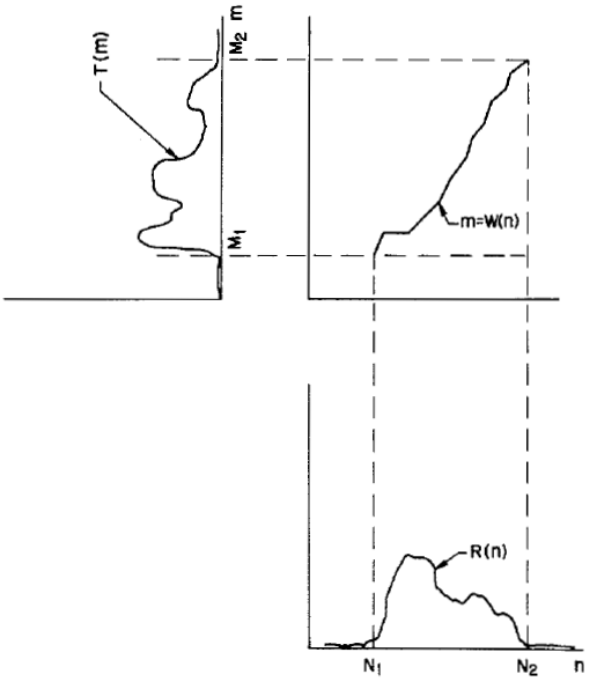
2.3.4 Inne metody klasyfikacji

Dynamiczne zwijanie czasowe DTW

Metoda dynamicznego zwijania czasowego DTW (ang. *Dynamic Time Warping*) została wymyślona pod koniec lat 80-tych [39], [40]. Stosowano ją do porównywania dwóch przebiegów czasowych o różnej długości. Metoda ta poza rozpoznawaniem mowy stosowana jest także w systemach wideo oraz grafice. Potrafi ona wychwycić podobieństwa np. w dwóch wypowiedziach tego samego słowa, nawet jeśli zostały wypowiedziane z różną szybkością. Obecnie odchodzi się od stosowania tej metody rozpoznawania, zastępując ją często ukrytymi modelami Markowa.

Problemem jest porównywanie dłuższych wzorców, gdzie znacznie rośnie nakład obliczeniowy oraz spada skuteczność działania algorytmu.

Najczęściej używaną miarą odległości w algorytmie DTW jest miara euklidesowa, choć możliwe jest stosowanie praktycznie dowolnej miary odległości. Proces optymalizacji korzysta z programowania dynamicznego (ang. *Dynamic Programmic – DP*), o czym świadczy także nazwa algorytmu „*Dynamic*" Time Warping.



Rys. 2.9 Ilustracja działania algorytmu DTW

Rys. 2.9 ilustruje sposób działania algorytmu DTW dla wektorów $R(n)$ i $T(m)$, dla których ścieżka (ang. *path*) lub funkcja zwijająca (ang. *warping function*) maksymalizuje podobieństwo między tymi dwoma sygnałami. Na Rys. 2.9 zaznaczono również początek słowa (N_1 dla $R(n)$ i M_1 dla $T(m)$) oraz jego koniec (N_2 dla $R(n)$ oraz M_2 dla $T(m)$). Bardziej szczegółowy opis metody DTW można znaleźć w [15 pp. 623-676].

Ukryte modele Markowa

Ukryte modele Markowa (ang. *The Hidden Markov Model - HMM*) [8], [14], [15], [23] są statystyczną metodą klasyfikacji sekwencji zdarzeń. Podstawy teoretyczne metody opracowane zostały pod koniec lat 60-tych przez Bauma i współpracowników [41], natomiast praktyczne zastosowanie HMM do rozpoznawania mowy należy przypisać Bakerowi z Uniwersytetu Carnegie-Mellon (CMU) [42] oraz Jelinkowi z firmy IBM [43], [44]. Ze względu na fakt, że ukryte modele Markowa nie będą wykorzystywane w sposób bezpośredni w pracy, pominięto szerszy opis tej metody.

2.3.5 Sztuczne sieci neuronowe

Sztuczne sieci neuronowe (ang. *Artificial Neural Network – ANN*) są pewną, bardzo uproszczoną formą naśladownictwa pracy układu nerwowego człowieka. Mimo że nie są idealne i nie mogą konkurować z rzeczywistymi neuronami, to posiadają szereg właściwości, które czynią je atrakcyjnymi dla potrzeb np. rozpoznawania. W opinii autora do najważniejszej zalety ANN należy jej **zdolność do generalizacji (uogólniania)**. Jest to cecha, której nie posiada np. technika ukrytych modeli Markowa.

Do innych zalet sztucznych sieci neuronowych należy zaliczyć:

- możliwość przetwarzania równoległego,
- odporność na uszkodzenia poszczególnych elementów sieci,
- elastyczność dzięki procesowi uczenia na różnych danych wejściowych,
- małe rozmiary i niski pobór mocy dla ich sprzętowej realizacji,
- umiejętność radzenia sobie z sygnałem zaszumionym lub niepełnym.

Niestety sztuczne sieci neuronowe nie są pozbawione także wad. Główne z nich to:

- brak możliwości porównywania sygnałów o różnej długości,
- możliwość „utknięcia" rozwiązania w minimum lokalnym,
- brak możliwości porównywania sygnałów niestacjonarnych.

Sztuczne sieci neuronowe znajdują szerokie zastosowanie w rozpoznawaniu i klasyfikacji wzorców, predykcji szeregów czasowych, analizie danych statystycznych, odszumianiu i kompresji sygnałów oraz w sterowaniu i automatyzacji.

Sztuczne sieci neuronowe stosowane są dziś m.in. do:

- rozpoznawania mowy i pisma (OCR),
- diagnostyki układów elektronicznych,
- badań psychiatrycznych i rekrutacji pracowników,
- prognozowania: sprzedaży, cen, zachowań giełdowych,
- badań geologicznych,
- analizy badań medycznych i biologicznych,
- planowania: remontów maszyn czy postępów w nauce,
- analizy problemów produkcyjnych,
- technikach kryminalistycznych,
- sterowaniu procesami przemysłowymi.

Sieci neuronowe ze względu na ich architekturę można w ogólności podzielić na trzy główne grupy: sieci jednokierunkowe, rekurencyjne oraz samoorganizujące się.

Sieci **jednokierunkowe** (ang. *feed-forward networks*), przekształcają sygnał wejściowy na wyjściowy wykorzystując odpowiednią funkcję. Żądane przekształcenie jest realizowane dzięki uczeniu z nauczycielem parametrów sieci. Sygnał przechodzi tylko raz przez każdy neuron w swoim cyklu. Najprostszą siecią neuronową jest perceptron progowy, opracowany przez McCullocha i Pittsa

[45]. Sieci jednokierunkowe ze względu na funkcję aktywacji można podzielić na **sigmoidalne** (perceptron wielowarstwowy) oraz **radialne** (sieci o radialnych funkcjach bazowych – ang. *Radial Basis Functions – RBF*).

W sieciach **rekurencyjnych** (ang. *feedback networks*) istnieje sprzężenie zwrotne między warstwami. Wspólną ich cechą jest przekazywanie wsteczne sygnałów z warstwy wyjściowej lub ukrytej do warstwy wejściowej. Przykładami sieci ze sprzężeniem są m.in. sieć autoasocjacyjna Hopfielda, sieć Hamminga, czy sieć typu BAM.

Sieci **samoorganizujące się** można podzielić na dwie grupy: działające na zasadzie współzawodnictwa oraz typu korelacyjnego. W pierwszej grupie sieci nazywanej samoorganizującymi się mapami (ang. *Self Organizing Maps – SOM*), lub od nazwiska ich autora - sieciami Kohonena [46], [47], [48]. Podstawą ich uczenia jest konkurencja między neuronami. W sieciach typu korelacyjnego mechanizm samoorganizacji oparty jest na regule asocjacji Hebba, wykorzystując współzależności między sygnałami. Ze względu na ten typ uczenia sieci tego typu zwane są również sieciami hebbowskimi.

Zastosowanie sieci neuronowych w rozpoznawaniu mowy – rys historyczny

W drugiej połowie lat 80-tych sieci neuronowe powróciły jako temat zainteresowań naukowców dzięki popularyzacji algorytmu wstecznej propagacji [49], który określa strategię doboru wag w sieci wielowarstwowej przy wykorzystaniu gradientowych metod optymalizacji. W tym okresie nastąpił znaczny rozwój zastosowań sztucznych sieci neuronowych do zagadnień związanych z rozpoznawaniem mowy. Jednak już wcześniej, bo w 1981 r., Kohonen wprowadził nowy typ sieci nazwany LVQ (ang. *Learning Vector Quantization*).

Pod koniec lat 80-tych i początku 90-tych zaczęto stosować ANN do rozpoznawania krótkich segmentów mowy takich jak fonemy. W tym okresie powstały sieci neuronowe z opóźnieniem czasowym TDNN (ang. *Time Delay Neural Network*) [10] oraz STLVQ (a*ng. LVQ-based Shift-Tolerant Neural Network*) [50], które przeważnie stosowano do rozpoznawania fonemów, osiągając dla tych sieci bardzo dobre efekty. Metody te nie nadawały się dla dłuższych segmentów mowy, takich jak słowa czy zdania. Wynikało to z tego, że struktura z opóźnieniem czasowym sprawdzała się tylko dla sygnałów stosunkowo krótkich, które można było uznać za stacjonarne (fonemy nadawały się do tego idealnie).

Sygnał mowy posiada strukturę czasową. To samo słowo może zostać wypowiedziane szybciej lub wolniej. Co więcej, zjawisko czasowego „marszczenia" (ang. *warping*) występuje w większym stopniu dla samogłosek niż dla spółgłosek. Można więc powiedzieć, że sygnał mowy jest w sposób nieliniowy „zwijalny" w czasie (ang. *nonlinearly time-warped*) [51]. Klasyczna sieć neuronowa nie jest jednak przystosowana do analizowania takiej struktury sygnału. Kolejnym krokiem było więc zastosowanie np. sieci TDNN, w połączeniu z takimi metodami jak DTW czy techniką ukrytych modeli Markowa, co pozwoliło na stworzenie systemów hybrydowych, które mają możliwość modelowania sygnałów niestacjonarnych charakteryzujących dłuższe sygnały [52].

W ostatnich latach powstały zupełnie nowego typu sieci neuronowe, takie jak np. tzw. *spiking networks* [53] czy technika wektorów podtrzymujących SVM. To właśnie sieciom SVM poświęcono cały rozdział 3.

3. Sieci SVM[3]

3.1 Wstęp do techniki wektorów podtrzymujących

Technika wektorów podtrzymujących nazywana też metodą wektorów nośnych, zwana dalej w skrócie SVM (ang. *Support Vector Machine*), należy do nowego typu technik uczenia z nauczycielem. Wraz z innymi metodami, takimi jak KFD (ang. *Kernel Fisher Discriminant*) [54], [55], [56], [57] czy KPCA (ang. *Kernel Principal Component Analysis*) [58], [59], [60] należy do technik uczenia maszynowego (ang. *machine learning*) wykorzystujących funkcje jądra.

Sieci neuronowe typu SVM wykorzystują specjalny typ uczenia, który maksymalizuje margines separacji między danymi przyporządkowanymi do dwóch różnych klas. Metoda ta w swej pierwotnej postaci została opracowana przez Vapnika [61], [62], [63].

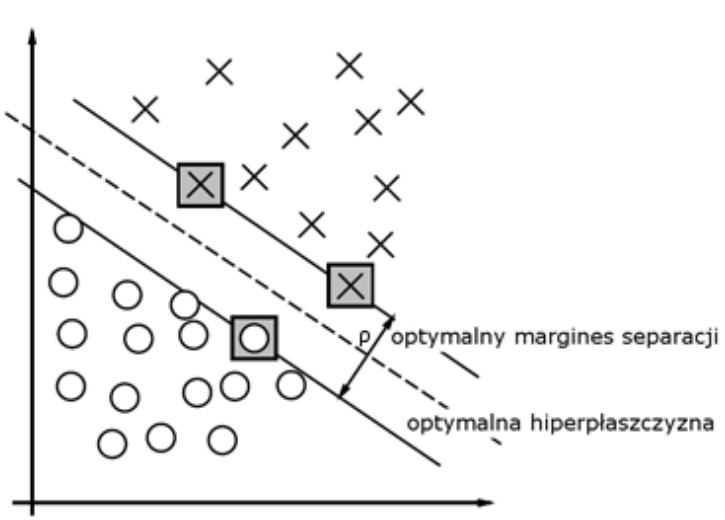
Metody wykorzystujące funkcje jądra, w tym SVM, zostały z powodzeniem zastosowane w wielu dziedzinach, np. w optycznym rozpoznawaniu wzorców i obiektów [64], [65], klasyfikacji tekstu [66], [67], w predykcji przebiegów czasowych [68], [69], analizie DNA i białek [70], w eliminacji zakłóceń w sieciach energetycznych [71], [72], problemach regresji [73] i wielu innych.

Sieci neuronowe typu SVM stosuje się także w zagadnieniach związanych z rozpoznawaniem mowy. Przykładem tego typu prac może być zastosowanie sieci SVM do klasyfikacji fonemów [74], [75] czy stworzenie hybrydowego systemu rozpoznawania mowy SVM/HMM [76]. W ostatnim czasie z powodzeniem stosowano SVM także do rozpoznawania cyfr, wypowiadanych przez wielu mówców w języku hiszpańskim [77].

Idea działania sieci typu SVM polega na potraktowaniu zbiorów danych wejściowych jako wektorów reprezentujących współrzędne punktów w przestrzeni wielowymiarowej. W tej przestrzeni konstruowana jest hiperpłaszczyzna dzieląca przestrzeń na dwie podprzestrzenie.

Z transformacją taką wiążą się pewne problemy: konceptualny i techniczny. Problem pierwszy dotyczy tego, w jaki sposób znaleźć hiperpłaszczyznę separującą o dobrych właściwościach generalizacyjnych. Wielowymiarowość przestrzeni cech będzie ogromna, a nie wszystkie hiperpłaszczyzny separują dane treningowe zachowując jednocześnie właściwości generalizacyjne. Techniczny problem polega natomiast na tym, jak w sposób efektywny dokonywać obliczeń w hiperprzestrzeni o bardzo dużej liczbie wymiarów. Przykładem może być skonstruowanie wielomianu 4. lub 5. stopnia w przestrzeni 200-wymiarowej, który może wymagać stworzenia hiperperpowierzchni w 10^9-wymiarowej przestrzeni cech [61].

[3] W rozdziale 3, podrozdz. od 3.2 do 3.5, zawierające opis teoretyczny techniki wektorów podtrzymujących, powstały głównie na podstawie [72] i [81].



Rys. 3.1 Przykład problemu separacji w 2-wymiarowej przestrzeni dla dwóch klas.

Dla separowalnych danych zostało znalezione rozwiązanie problemu konceptualnego, czyli równania optymalnej hiperpłaszczyzny [78]. Optymalna hiperpłaszczyzna na Rys. 3.1 [61] jest zdefiniowana jako liniowa funkcja decyzyjna z maksymalnym marginesem separacji między wektorami obu klas. Zaobserwowano, że konstruując taką hiperpłaszczyznę należy użyć tylko niektórych wybranych wektorów treningowych tzw. wektorów podtrzymujących (ang. *support vectors*), które definiują margines separacji. Na Rys. 3.1 zaznaczono je szarymi kwadratami.

Tab. 3.1 Tabela podobieństw i różnic pomiędzy tradycyjnymi sieciami neuronowymi a sieciami SVM

Klasyczna sieć neuronowa	**Sieć neuronowa typu SVM**
Minimalizowana funkcja celu może posiadać minima lokalne, w których może utknąć proces uczenia.	Dzięki wykorzystaniu metod programowania nieliniowego gwarantowane jest osiągnięcie minimum globalnego.
Algorytm uczący nie jest w stanie modyfikować złożoności sieci neuronowej.	Dzięki zmianie parametru C (3.26), algorytm uczący jest w stanie kontrolować wzajemną zależność pomiędzy złożonością reguły decyzyjnej a częstotliwością występowania błędów [61].
Możliwe jest bezpośrednie wykorzystanie funkcji nieliniowych. Występuje zależność złożoności obliczeniowej od liczby neuronów ukrytych sieci. Istnieje problem tzw. przekleństwa wielowymiarowości (ang. *curse of dimensionality*) [79].	Istnieje możliwość wykorzystania zalet funkcji jądra. Złożoność obliczeniowa nie jest zależna od przestrzeni wejściowej **x**, ani od liczby cech K. Zależy ona jedynie od liczby danych uczących p.
Istnieje możliwość wyboru spośród wielu klas.	Istnieje możliwość wyboru tylko spośród dwóch klas. Aby zastosować sieć do rozpoznawania wielu klas, stosuje się metody wielokrotnej klasyfikacji.

Sieci typu SVM w niektórych aspektach są podobne do klasycznych sieci neuronowych. Zestawienie podobieństw i różnic przedstawiono w Tab. 3.1. Podobnie jak sieci klasyczne, sieci SVM podlegają uczeniu na zbiorze danych uczących oraz stosują podobne funkcje aktywacji. W przeciwieństwie jednak do klasycznych technik uczenia sieci neuronowych, dzięki wykorzystaniu metod programowania nieliniowego, proces uczenia praktycznie zawsze prowadzi do znalezienia minimum globalnego funkcji błędu [80]. Kolejną zaletą sieci typu SVM jest możliwość kontroli przez algorytm złożoności sieci neuronowej. Dzięki zastąpieniu funkcji aktywacji neuronów funkcją jądra $K(\mathbf{x}_i, \mathbf{x})$, stała się możliwa znaczna redukcja operacji matematycznych zarówno w trybie uczenia, jak i odtwarzania. Dzięki uniezależnieniu złożoności obliczeniowej od liczby cech, czyli liczby neuronów ukrytych sieci, uniknięto tzw. przekleństwa wielowymiarowości.

Sieci SVM można zakwalifikować do grupy sieci jednokierunkowych, które zwykle mają strukturę dwuwarstwową, tzn. zawierają w sobie 1 warstwę ukrytą oraz warstwę wyjściową. Dzięki funkcji jądra można stosować różne typy funkcji aktywacji, w tym funkcje: liniową, wielomianową, sigmoidalną czy radialną.

Przy uczeniu sieci typu SVM, podobnie jak ma to miejsce w przypadku klasycznych sieci neuronowych, wykorzystywane są dane uczące, stanowiące parę $\mathbf{x}_i, d_i$, $i = 1, 2, ..., p$, gdzie $\mathbf{x}_i$, jest wektorem wejściowym, a d_i - daną klasą (w przypadku dwóch klas $d_i = -1$ lub $d_i = 1$). W ogólności $\boldsymbol{d}$ tworzy tzw. wektor celu (ang. *target vector*). Zawiera on informacje, do których klas przynależą dane wejściowe. Po etapie uczenia następuje zamrożenie parametrów sieci. Kolejnym etapem jest możliwość pracy sieci w trybie odtworzeniowym. Na wejście podaje się sygnały wejściowe, a sieć samodzielnie już, na podstawie wcześniej przeprowadzonego etapu uczenia, przyporządkowuje dane do odpowiednich klas. Na etapie klasyfikacji uwidacznia się kolejna różnica między klasycznymi sieciami a sieciami typu SVM polegająca na tym, że w tych ostatnich klasyfikacja odbywa się tylko między dwiema klasami. W przypadku większej liczby klas konieczne jest zastosowanie wielokrotnej klasyfikacji typu:

- jeden przeciw wszystkim (ang. *one vs. all*),
- jeden przeciw jednemu (ang. *one vs. one*),
- bądź ich kombinacji.

Rozpoznawanie wielu klas zostanie szerzej omówione w podrozdz. 3.7.

3.2 Rozwiązanie problemu klasyfikacji wzorców liniowo separowanych z użyciem sieci SVM

Dla zbioru danych uczących $\mathbf{x}_i, d_i$, $i = 1, 2, ..., p$, problem separowalności [81], [82] można sformułować jako

$$\begin{aligned} \mathbf{w}^T \mathbf{x}_i + b > 0 \quad &\rightarrow \quad d_i = 1 \\ \mathbf{w}^T \mathbf{x}_i + b < 0 \quad &\rightarrow \quad d_i = -1 \end{aligned} . \tag{3.1}$$

Zakładając liniową separowalność obu klas, równanie płaszczyzny separującej ma postać

$$g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = 0 . \tag{3.2}$$

W równaniu (3.2) wektor wejściowy jest N-wymiarowy. Wtedy wektor wagowy $\mathbf{w}$ jest także N-wymiarowy. Zgodnie z [80], [63], za optymalną hiperpłaszczyznę separującą

$$g(\mathbf{x}) = \mathbf{w}_0^T \mathbf{x} + b_0 , \tag{3.3}$$

uważana jest taka hiperpłaszczyzna, która maksymalizuje margines separacji ρ (patrz Rys. 3.1).

Odległość r dowolnego punktu $\mathbf{x}$ w przestrzeni wielowymiarowej od optymalnej hiperpłaszczyzny wynosi [63]

$$r\ \mathbf{x}\ = \frac{g\ \mathbf{x}}{\|\mathbf{w}_0\|}, \tag{3.4}$$

gdzie $\|\mathbf{w}_0\|$ jest normą euklidesową wektora $\mathbf{w}_0$ definiowaną jako

$$\|\mathbf{w}_0\| = \left\| \left[w_{0_1}, w_{0_2}, ..., w_{0_N} \right] \right\| = \sqrt{\left| w_{0_1}{}^2 \right| + \left| w_{0_2}{}^2 \right| + ... + \left| w_{0_N}{}^2 \right|} = \sqrt{\sum_{i=1}^{N} w_{0_i}{}^2} . \tag{3.5}$$

Wtedy odległość początku układu współrzędnych od tej hiperpłaszczyzny jest szczególnym przypadkiem $r(\mathbf{x})$ dla $\mathbf{x} = 0$.

$$r\ \mathbf{0}\ = \frac{g\ \mathbf{0}}{\|\mathbf{w}_0\|} = \frac{b_0}{\|\mathbf{w}_0\|} . \tag{3.6}$$

W praktyce bardzo często wartość zerowa po prawej stronie nierówności w (3.1) zastępowana jest wartością tolerancji ε . Nowo otrzymane nierówności nieostre normalizuje się tak, aby prawa strona była równa 1. Zastępując zmodyfikowane wartości wag nowymi oznaczeniami, warunki przypisania do obu klas przybierają postać [81], [63]

$$\begin{aligned} \mathbf{w}_0^T \mathbf{x}_i + b_0 \geq 1 \ &\rightarrow\ d_i = 1 \\ \mathbf{w}_0^T \mathbf{x}_i + b_0 \leq -1 \ &\rightarrow\ d_i = -1 \end{aligned} . \tag{3.7}$$

Jeśli para punktów $\mathbf{x}_i, d_i$ spełnia (3.7) ze znakiem równości, to wektor $\mathbf{x}_i$ tworzy wektor podtrzymujący [61]. Wektory podtrzymujące są tymi punktami danych, które leżą najbliżej optymalnej hiperpłaszczyzny i są jednocześnie najtrudniejsze do klasyfikacji. Tylko one decydują o położeniu hiperpłaszczyzny oraz marginesie separacji, gdyż wszystkie punkty leżące poza nimi spełniają warunek prawidłowej klasyfikacji z nadmiarem. Dla wektorów podtrzymujących $\mathbf{x}_s$ spełniony jest warunek

$$g\ \mathbf{x}_s\ = \mathbf{w}_0^T \mathbf{x}_s \pm b_0 = \pm 1 \tag{3.8}$$

dla $d_s = \pm 1$. Odległość wektorów podtrzymujących od optymalnej hiperpłaszczyzny dana jest jako

$$r = \frac{g(\mathbf{x}_s)}{\|\mathbf{w}_0\|} = \begin{cases} \dfrac{1}{\|\mathbf{w}_0\|} \text{ dla } d_s = 1 \\ -\dfrac{1}{\|\mathbf{w}_0\|} \text{ dla } d_s = -1 \end{cases}. \tag{3.9}$$

Wtedy margines separacji ρ pomiędzy dwiema klasami można przedstawić jako równy podwójnej wartości r

$$\rho = 2r = \frac{2}{\|\mathbf{w}_0\|}. \tag{3.10}$$

Aby uzyskać dobre wyniki klasyfikacji, niepodlegające wpływom zaszumienia czy zniekształcenia danych, pożądana jest jak największa wartość marginesu separacji ρ. Z (3.10) widać, że maksymalizacja marginesu separacji między dwoma klasami jest równoważna minimalizacji normy euklidesowej wektora $\mathbf{w}_0$.

Problem uczenia liniowych sieci SVM, czyli optymalnego doboru wektora wagowego $\mathbf{w}_0$ oraz wartości odniesienia (ang. *bias*) b_0 dla danych liniowo separowalnych, sprowadza się do maksymalizacji marginesu separacji przy spełnieniu warunku (3.7). Nosi on nazwę problemu pierwotnego *(*ang. *primal problem*) [63], [68], [81] i zapisywany jest jako

$$\min_{\mathbf{w}} \left(\frac{1}{2} \mathbf{w}^T \mathbf{w} \right), \tag{3.11}$$

przy ograniczeniu

$$d_i \left(\mathbf{w}^T \mathbf{x}_i + b \right) \geq 1. \tag{3.12}$$

Jest to problem tzw. programowania kwadratowego z liniowymi ograniczeniami względem wag. Jego rozwiązanie można otrzymać metodą mnożników Lagrange'a [83] (patrz także Dodatek A), dzięki minimalizacji funkcji Lagrange'a

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2} \mathbf{w}^T \mathbf{w} - \sum_{i=1}^{p} \alpha_i \left[d_i \left(\mathbf{w}^T \mathbf{x}_i + b \right) - 1 \right]. \tag{3.13}$$

W funkcji tej wektor mnożników Lagrange'a $\boldsymbol{\alpha}$ składa się z poszczególnych mnożników $\alpha_i > 0$. Rozwiązanie problemu minimalizacji funkcji Lagrange'a względem optymalizowanych parametrów $\mathbf{w}, b, \boldsymbol{\alpha}$ jest określone przez punkt siodłowy funkcji $L(\mathbf{w}, b, \boldsymbol{\alpha})$ i wyznaczane przez jej minimalizację względem $\mathbf{w}$ i b oraz maksymalizację względem wszystkich wartości α_i. Warunki optymalności rozwiązania względem $\mathbf{w}$ i b określają wzory

$$\frac{\partial L(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial \mathbf{w}} = \mathbf{0} \quad \rightarrow \quad \mathbf{w} - \sum_{i=1}^{p} \alpha_i d_i x_i = \mathbf{0} \tag{3.14}$$

oraz

$$\frac{\partial L(\mathbf{w}, b, \alpha)}{\partial b} = \mathbf{0} \quad \rightarrow \quad \sum_{i=1}^{p} \alpha_i d_i = \mathbf{0}. \tag{3.15}$$

Rozwiązanie układu równań (3.14) i (3.15) przyjmuje postać

$$\mathbf{w} = \sum_{i=1}^{p} \alpha_i d_i x_i \,. \tag{3.16}$$

Dla wyznaczenia wartości b wykorzystywany jest fakt, że w punkcie siodłowym iloczyn mnożnika przez odpowiednie ograniczenie znika. Po wstawieniu zależności (3.14), (3.15) i (3.16) do wzoru na funkcję Lagrange'a w punkcie rozwiązania otrzyma się zgodnie z [81], [68] zależność

$$L\ \ \mathbf{w}, b, \boldsymbol{\alpha}\ \ = Q\ \ \boldsymbol{\alpha}\ \ = \sum_{i=1}^{p} \alpha_i - \frac{1}{2} \sum_{i=1}^{p} \sum_{j=1}^{p} \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j \,. \tag{3.17}$$

Proces uczenia sieci polega na maksymalizacji funkcji $Q(\alpha)$ względem mnożników Lagrange'a. Określony przez równanie (3.11) problem pierwotny zamienia się w problem dualny *(*ang. *dual problem)*, który sformułować można jako [81], [63], [68]

$$\max_{\alpha_i} Q(\alpha) = \max_{\alpha_i} \left(\sum_{i=1}^{p} \alpha_i - \frac{1}{2} \sum_{i=1}^{p} \sum_{j=1}^{p} \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j \right), \tag{3.18}$$

gdzie

$$\alpha_i \geq 0 \tag{3.19}$$

oraz

$$\sum_{i=1}^{p} \alpha_i d_i = 0 \,, \tag{3.20}$$

dla $i = 1,2,\ldots,p$. Rozwiązanie problemu optymalizacyjnego opisanego wzorem (3.18) względem mnożników Lagrange'a $\alpha_i = \alpha_{i0}$ umożliwia wyznaczenie równania optymalnej hiperpłaszczyzny

$$\mathbf{w}_0 = \sum_{i=1}^{p} \alpha_{i0} d_i \mathbf{x}_i \,. \tag{3.21}$$

Wartość b_0 wyznaczana jest bezpośrednio z równania hiperpłaszczyzny (3.21) dla dowolnego wektora podtrzymującego $\mathbf{x} = \mathbf{x}_{si}$, z którego wynika, że

$$\mathbf{w}^T \mathbf{x}_{si} + b_0 = 1 \ \ \rightarrow \ \ b_0 = 1 - \mathbf{w}_0^T \mathbf{x}_{si} \,. \tag{3.22}$$

Mnożniki Lagrange'a, które otrzymuje się z rozwiązania zadania optymalizacyjnego (3.18), (3.19) i (3.20), spełniają warunek mówiący, że iloczyn mnożnika przez wartość funkcji ograniczenia dla każdej pary danych uczących jest równy zeru. Oznacza to, że wszystkie mnożniki, dla których ograniczenie jest nieaktywne (znak większości bądź mniejszości) muszą być równe zeru. Niezerowe wartości mnożników występują jedynie dla wektorów podtrzymujących, dla których ograniczenie jest typu aktywnego (znak równości). Stąd wzór określający optymalną wartość wektora $\mathbf{w}_0$ może zostać uproszczony do postaci

$$\mathbf{w}_0 = \sum_{i=1}^{Ns} \alpha_{si} d_{si} \mathbf{x}_{si} \,, \tag{3.23}$$

w której s oznacza przynależność do zbioru wektorów podtrzymujących, a N_s jest liczebnością zbioru. Wartość stałej b_0 zostaje określona na podstawie (3.20), przy wyborze dowolnego wektora podtrzymującego $\mathbf{x}_{si}$.

Z zależności (3.23) wynika, że równanie optymalnej hiperpłaszczyzny zależy wyłącznie od wektorów podtrzymujących, a pozostałe wektory ze zbioru danych uczących nie mają wpływu na wynik rozwiązania. Jest to jedna z większych zalet sieci SVM.

3.3 Rozwiązanie problemu klasyfikacji wzorców liniowo nieseparowalnych z użyciem liniowej sieci SVM

Bardzo często przy klasyfikacji spotyka się wzorce, których nie można rozdzielić linią prostą lub hiperpłaszczyzną. Wzorce takie określa się wtedy jako liniowo nieseparowane. Przy rozwiązywaniu problemu klasyfikacji takich wzorców, należy określić optymalną hiperpowierzchnię, która będzie minimalizować prawdopodobieństwo błędu klasyfikacji na zbiorze danych uczących, zachowując jednocześnie możliwie najszerszy margines separacji. W takim przypadku dane mogą przekroczyć strefę marginesu separacji na dwa sposoby. W pierwszym przypadku punkt danych $(\mathbf{x}_i, d_i)$ jest położony wewnątrz lub na zewnątrz strefy, ale po niewłaściwej stronie optymalnej hiperpłaszczyzny, co odpowiada wystąpieniu błędu klasyfikacji. W drugim przypadku punkt danych $(\mathbf{x}_i, d_i)$ jest położony wewnątrz strefy, ale po właściwej stronie optymalnej hiperpłaszczyzny. Nie występuje wówczas błąd klasyfikacji, a jedynie zmniejszenie wartości marginesu.

Przy rozwiązywaniu problemu klasyfikacji danych nieseparowalnych liniowo ograniczenia, o których mowa w (3.12), można zapisać w postaci uogólnionej

$$d_i \ \mathbf{w}_i^T \mathbf{x}_i + b \ \geq 1 - \xi_i, \tag{3.24}$$

w której $\xi_i > 0$ jest zmienną dopełniającą. Jej wartość efektywnie zmniejsza margines separacji, aż do jego przekroczenia. Wektory podtrzymujące spełniają równanie

$$d_i \ \mathbf{w}_i^T \mathbf{x}_i + b \ = 1 - \xi_i . \tag{3.25}$$

Uwzględnianie tych wektorów, łącznie z wektorami prowadzącymi do błędnej klasyfikacji, wpływa na położenie optymalnej hiperpłaszczyzny.

Przy klasyfikacji wzorców nieseparowalnych liniowo, problem pierwotny można sprowadzić do minimalizacji zmodyfikowanej funkcji celu $\phi \ \mathbf{w}, \xi$ zgodnie z [81], [68] i zapisać jako

$$\phi \ \mathbf{w}, \xi \ = \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^{p} \xi_i , \tag{3.26}$$

przy ograniczeniu (3.23). Czynnik pierwszy we wzorze (4.25) dotyczy maksymalizacji marginesu separacji, natomiast drugi odpowiada za przyszłe błędy testowania. Suma $\sum_{i=1}^{p} \xi_i$ jest maksymalnym górnym oszacowaniem liczby tych błędów. Parametr $\boldsymbol{C}$, nazywany czasem pojemnością, jest wagą, z jaką traktowane są błędy testowania w stosunku do marginesu separacji, decydującym o złożoności przyszłej sieci neuronowej [84], [63]. Jest to parametr regulujący, który użytkownik dobiera w sposób eksperymentalny w czasie treningu połączonego z bieżącą oceną wyników, np. metodą *grid-search* opisaną w podrozdz. 3.8. Parametr $\boldsymbol{C}$ ma duży wpływ na błąd, stąd też jego wartość należy dobierać ostrożnie, ze względu na niebezpieczeństwo nadmiernego dopasowania modelu do danych uczących.

Podobnie jak ma to miejsce w przypadku wzorców separowalnych liniowo, problem pierwotny przy klasyfikacji wzorców liniowo nieseparowalnych sprowadzany jest do problemu dualnego [81], [63].

Dla danego zbioru danych uczących $(\mathbf{x}_i, d_i)$ określa się mnożniki Lagrange'a α_i, które maksymalizują wartość funkcji błędu $Q(\alpha)$

$$\max_{\alpha_i} Q(\alpha) = \max_{\alpha_i} \left(\sum_{i=1}^{p} \alpha_i - \frac{1}{2} \sum_{i=1}^{p} \sum_{j=1}^{p} \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j \right), \tag{3.27}$$

przy ograniczeniach:

oraz

$$\sum_{i=1}^{p} \alpha_i d_i = 0 \qquad 0 \le \alpha_i \le C \tag{3.28}$$

dla $i = 1,2,\ldots,p$ i stałej C wybranej przez użytkownika [63].

Ze sformułowania problemu dualnego wynika, że ani zmienne dopełniające, ani też mnożniki Lagrange'a z nią związane nie pojawiają się w sposób jawny w sformułowaniu problemu. Jedyną różnicą w stosunku do separowalności liniowej jest zmiana w ograniczeniu górnym mnożników Lagrange'a. Zamiast warunku $\alpha_i > 0$ (3.19) jest $0 \le \alpha_i \le C$ (3.28).

Po rozwiązaniu problemu dualnego otrzymuje się wyrażenie na wektor $\mathbf{w}_0$ optymalnej hiperpłaszczyzny w postaci takiej samej jak w przypadku separowalności liniowej wzorców

$$\mathbf{w}_0 = \sum_{i=1}^{Ns} \alpha_{si} d_{si} \mathbf{x}_{si} \,. \tag{3.29}$$

Aby określić wartości współczynnika b_0 należy wybrać dowolny wektor podtrzymujący $\mathbf{x}_{si}$, dla którego $0 \le \alpha_i \le C$ oraz $\xi_{si} = 0$, oraz skorzystać ze wzoru (3.22).

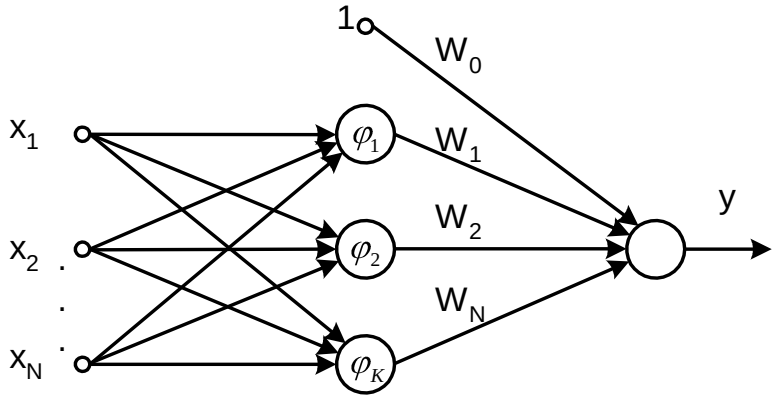
3.4 Struktura nieliniowej sieci SVM

Nieseparowalność liniowa jest ograniczeniem separowalności, które można wyeliminować stosując egzystencjalne twierdzenia Covera [85]. Mówi ono, że wzorce nieseparowalne liniowo można w sposób nieliniowy przetransformować w wielowymiarową przestrzeń cech (parametrów) o wymiarze $K > N$, gdzie z dużym prawdopodobieństwem będą one separowalne liniowo.

Zakładając, że $\mathbf{x}$ oznacza wektor wejściowy opisujący proces, to po zrzutowaniu go w przestrzeń K-wymiarową jest on reprezentowany przez zbiór cech $\phi_j(\mathbf{x})$ [86], dla $j = 1,2,\ldots,K$. Równanie hiperpłaszczyzny z przestrzeni liniowej (3.2) może być wtedy zastąpione przez równanie hiperpłaszczyzny w przestrzeni cech

$$\mathbf{w}^T \phi(\mathbf{x}) + b = \sum_{j=1}^{K} w_j \varphi_j(\mathbf{x}) + b, \tag{3.30}$$

w której wagi w_j tworzą zbiór wag pochodzących od $\varphi_j\ \mathbf{x}$ [82], [87] do neuronu wyjściowego, a $w_0 = b$ jest wagą polaryzacji. Równanie (3.30) jest wtedy równaniem hiperpowierzchni w przestrzeni x. Cechy procesu opisane funkcjami $\varphi_j\ \mathbf{x}$ przejmują w takim przypadku rolę poszczególnych zmiennych x_j.



Rys. 3.2 Struktura sieci neuronowej opartej na radialnych funkcjach bazowych

Do równania (3.30) można przyporządkować strukturę sieci neuronowej analogiczną sieci opartej na radialnych funkcjach bazowych (ang. *Radial Basis Functions - RBF*), którą przedstawiono na Rys. 3.2. Podstawową różnicą w stosunku do sieci RBF stanowi fakt, że funkcje $\varphi_j\ \mathbf{x}$ mogą być dowolnej postaci [81].

Przyjmując, że

$$\boldsymbol{\varphi}\ \mathbf{x} = \left[1, \varphi_1\ \mathbf{x}\ , ..., \varphi_K\ \mathbf{x}\ \right]^T \tag{3.31}$$

oraz

$$\mathbf{w} = \left[w_0, w_1, ..., w_K\right]^T, \tag{3.32}$$

to równanie hiperpłaszczyzny w przestrzeni cech dane jest przez

$$\mathbf{w}^T \boldsymbol{\varphi}\ \mathbf{x} = 0. \tag{3.33}$$

Zakładając liniową separowalność wzorców w przestrzeni cech można uogólnić rozwiązanie optymalnej hiperpłaszczyzny z przestrzeni $\mathbf{x}$, opisanej wzorem (3.29) na przestrzeń cech

$$\mathbf{w} = \sum_{i=1}^{p} \alpha_i d_i \varphi\ \mathbf{x}_i\ . \tag{3.34}$$

Wprowadzając wyrażenie na wektor $\mathbf{w}$ do wzoru hiperpłaszczyzny otrzymuje się

$$\mathbf{w}^T \boldsymbol{\varphi}\ \mathbf{x} = \sum_{i=1}^{p} \alpha_i d_i \boldsymbol{\varphi}^T\ \mathbf{x}_i\ \boldsymbol{\varphi}\ \mathbf{x} = \mathbf{0}. \tag{3.35}$$

Iloczyn skalarny wektorów $\boldsymbol{\varphi}\ \mathbf{x}$ i $\boldsymbol{\varphi}\ \mathbf{x}_i$, gdzie indeks i oznacza dane treningowe, można traktować jako funkcje jądra $K\ \mathbf{x}_i, \mathbf{x}$ zgodnie z [82], [63], [88] i zapisać w postaci

$$K(\mathbf{x},\mathbf{x}_i) = \boldsymbol{\varphi}^T(\mathbf{x})\,\boldsymbol{\varphi}(\mathbf{x}_i) = \sum_{j=0}^{K} \varphi_j(\mathbf{x})\,\varphi_j(\mathbf{x}_i) \quad . \tag{3.36}$$

Korzystając z faktu, że jądro $K(\mathbf{x}_i,\mathbf{x})$ jest funkcją symetryczną, tzn. $K(\mathbf{x}_i,\mathbf{x}) = K(\mathbf{x},\mathbf{x}_i)$, równanie hiperpłaszczyzny (3.35) można zapisać zgodnie z [81] jako

$$\sum_{i=1}^{p} \alpha_i d_i K(\mathbf{x},\mathbf{x}_i) = 0 \,. \tag{3.37}$$

Aby $K(\mathbf{x}_i,\mathbf{x})$ mogło być jądrem i zostać użyte w sieci SVM, musi spełniać następujące warunki określone w twierdzeniu Mercera [89], [63], [90], tzn. być funkcją symetryczną, spełniającą warunek iloczynu skalarnego dwóch bliżej nieokreślonych funkcji wektorowych $\boldsymbol{\varphi}(\mathbf{x})$ i $\boldsymbol{\varphi}(\mathbf{x}_i)$, a funkcja, która je spełnia, nazywana jest jądrem Mercera.

W Tab. 3.2 przedstawiono zależności odnoszące się do najczęściej używanych typów jąder: wielomianowego, radialnego oraz sigmoidalnego [91].

Tab. 3.2 Wybrane typy jąder [4]

Typ jądra	**Równanie**	**Uwagi**
liniowe	$K(\mathbf{x}_i,\mathbf{x}) = \mathbf{x}^T\mathbf{x}_i$	
wielomianowe	$K(\mathbf{x}_i,\mathbf{x}) = (\gamma\mathbf{x}^T\mathbf{x}_i + r)^p$	p dowolne, $\gamma > 0$
radialne	$K(\mathbf{x}_i,\mathbf{x}) = \exp(-\gamma\lVert\mathbf{x}-\mathbf{x}_i\rVert^2)$	sieć o jednej warstwie ukrytej i jednym neuronie wyjściowym, $\gamma > 0$
sigmoidalne	$K(\mathbf{x}_i,\mathbf{x}) = th(\gamma\mathbf{x}^T\mathbf{x}_i + r)$	sieć o jednej warstwie ukrytej i jednym neuronie wyjściowym, ograniczenia na γ i r

W sieciach z jądrem typu radialnego, liczba funkcji radialnych i ich centra są automatycznie utożsamiane z wektorami podtrzymującymi.

3.5 Własności nieliniowej sieci SVM

W początkowym etapie budowania sieci typu SVM pierwotna liczba wektorów podtrzymujących jest bardzo duża i zwykle równa jest ona ilości danych treningowych. W czasie procesu uczenia, w zależności od założonych parametrów (np. wartość parametru C dla sieci typu C-SVC (ang. *C-Support Vector Classification*) [84], [87], [63]), pierwotna struktura sieci jest redukowana i tylko wybrana część wcześniejszych wektorów pozostaje nadal wektorami podtrzymującymi.

[4] W Tab. 3.2 γ, r oraz p oznaczają parametry funkcji jądra.

W procesie treningu (uczenia) sieci SVM korzysta się najczęściej z metod programowania kwadratowego przy linowych ograniczeniach [82]. Metody te prowadzą zawsze do minimum globalnego, co jest główną różnicą w stosunku do klasycznych metod klasycznego uczenia sieci, w których często proces uczenia utykał w minimum lokalnym [81].

Kolejną zaletą sieci SVM jest to, że rozmiar problemu optymalizacyjnego nie zależy od wymiaru przestrzeni wejściowej $\mathbf{x}$, ani od liczby cech K. Rozmiar ten jest jedynie uzależniony od liczby danych uczących p. W procesie uczenia, jak i na etapie odtwarzania, sieć nie korzysta z informacji zawartych w funkcji $\boldsymbol{\varphi}(\mathbf{x})$, a jedynie zależy od jądra $K\ \mathbf{x}_i, \mathbf{x}$ [82]. Dzięki temu stosunkowo prosto można uniknąć problemów związanych z wielowymiarowością, np. przekleństwa wielowymiarowości.

W trybie odtwarzania nie ma także potrzeby jawnego obliczenia wartości $\boldsymbol{\varphi}(\mathbf{x})$, a jedynie $K\ \mathbf{x}_{si}, \mathbf{x}$, co znacznie przyspiesza procesy obliczeniowe.

$$y\ \mathbf{x} = \sum_{i=1}^{Ns} \alpha_{0i} d_i K\ \mathbf{x}_{si}, \mathbf{x} + b_0 \tag{3.38}$$

We wzorze (3.37) indeks s_i oznacza wektor podtrzymujący (ang. *support vector*) z grupy wektorów treningowych $\mathbf{x}_i$.

Jedną z ważniejszych zalet sieci SVM jest możliwość płynnej kontroli stopnia jej złożoności, która jest wyrażona miarą Vapnika-Chervonenkisa *VCdim* [84], [63]. Możliwe jest to dzięki możliwości regulacji szerokością marginesu separacji.

3.6 Sieć typu One-class SVM

W 2001 roku [92] zaproponowano nowy typ sieci SVM, zaprojektowany pierwotnie do estymacji rozkładów w przestrzeniach wielowymiarowych. W tego typu sieciach na etapie uczenia zakłada się, że wszystkie wektory treningowe $\mathbf{x}_i \in \mathbb{R}^n, i = 1, \ldots, p$ należą do jednej klasy, stąd też nazwa „*one-class*". Zbędna jest więc informacja, jak ma to miejsce w przypadku innych sieci SVM, o przynależności do konkretnej klasy.

Po nauczeniu sieci można podawać na wejście sieci dane należące do różnych klas. Mimo to sieć podejmie decyzję binarną, czyli czy dane na wejściu należą do klasy, na której była uczona sieć, czy też do niej nie należą.

Problem pierwotny w sieciach one-class SVM definiowany jest [92], [84] jako

$$\min_{w,b,\xi,\rho} \left[\frac{1}{2} (\mathbf{w}^i)^T \mathbf{w}^i - \rho + \frac{1}{\nu p} \sum_{i=1}^{p} \xi_i \right], \tag{3.39}$$

gdzie $v \in (0,1]$ i przy ograniczeniach:

$$\mathbf{w}^T \phi(\mathbf{x}_i) \geq \rho - \xi_i, \tag{3.40}$$

oraz

$$\xi_i \geq 0, \tag{3.41}$$

dla wszystkich $i = 1, \ldots, p$.

Problem dualny sprowadza się do minimalizacji:

$$\min_{\boldsymbol{\alpha}} \left[\frac{1}{2} \boldsymbol{\alpha}^T Q \boldsymbol{\alpha} \right], \tag{3.42}$$

gdzie:

$$Q_{ij} = K(x_i, x_j) \equiv \phi(x_i)^T \phi(x_j), \tag{3.43}$$

przy ograniczeniach:

$$0 \leq \alpha_i \leq \frac{1}{vp}, \tag{3.44}$$

oraz

$$\mathbf{e}^T \boldsymbol{\alpha} = 1. \tag{3.45}$$

Funkcja decyzyjna, przyjmująca wartości -1 lub 1, definiowana jest jako

$$\operatorname{sgn}\left(\sum_{i=1}^{p} \alpha_i K(\mathbf{x}_i, \mathbf{x}) - \rho\right), \tag{3.46}$$

gdzie:

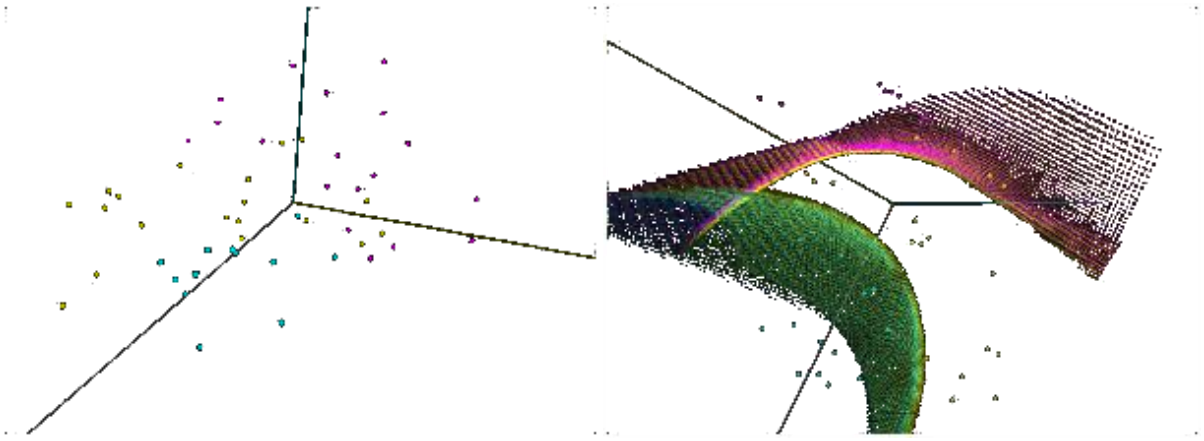
$$\operatorname{sgn}(x) = \begin{cases} 1 \, dla \, x \geq 0 \\ -1 \, dla \, x < 0 \end{cases}. \tag{3.47}$$

Dobór optymalnych parametrów sieci polega na regulacji parametru v oraz parametrów jądra, które mogą być takie jak w innych typach sieci SVM (patrz Tab. 3.2). Stwierdzono, że wraz ze wzrostem parametru v zwiększa się liczba wektorów podtrzymujących oraz wydłuża się czas uczenia sieci [92].

Sieci typu one-class SVM są szczególnie przydatne w sytuacjach, kiedy potrzeba wykryć nowość w sygnale, a nie jest potrzebna do tego znajomość wszystkich danych z modelu. Działają szybciej i mają mniejsze wymagania pamięciowe, ponieważ zawierają mniej danych potrzebnych do uczenia sieci.

3.7 Rozpoznawanie wielu klas w sieci SVM

Sieci SVM zostały pierwotnie zaprojektowane do binarnej klasyfikacji danych [87], dokonując początkowo rozdziału między dwoma klasami danych. Rozpoznawanie wielu klas z użyciem sieci SVM wymaga stosowania wielokrotnej klasyfikacji [68].



Rys. 3.3 Ilustracja problemu klasyfikacji w przestrzeni o trzech wymiarach

Na Rys. 3.3 po lewej stronie widoczne są zbiory punktów w trzech barwach. Poszczególne kolory reprezentują przynależność punktu do danej klasy. Każdy z punktów odpowiada wektorowi parametrów, na który składają się współrzędne w przestrzeni. W tym prostym przypadku jest to przestrzeń trójwymiarowa, więc każdy z wektorów ma wymiar 3. Z prawej strony przedstawiono podział hiperpłaszczyznami, aby rozdzielić każdą z klas. Jest to bardzo uproszczony przykład klasyfikacji z użyciem sieci SVM. W rzeczywistości w niniejszej pracy do rozpoznawania mowy obliczenia dokonywane są w przestrzeniach *k*-wymiarowych, w których *k* równe jest długości wektorów parametrów. Biorąc pod uwagę, że w niniejszej pracy autor korzysta z 10 parametrów na ramkę, a liczba ramek wynosi średnio kilkadziesiąt, oznacza to, że sieć SVM pracuje w hiperprzestrzeniach o kilkuset wymiarach. Co więcej, w takiej hiperprzestrzeni sieć SVM musi znaleźć optymalne hiperpowierzchnie, które będą rozgraniczać z jak największym marginesem separacji kilkadziesiąt klas.

3.7.1 Algorytm typu „jeden przeciw wszystkim”

Do jednej z najbardziej znanych należy metoda *„jeden przeciw wszystkim*” [81], [87]. Prawdopodobnie pierwsze praktyczne wykorzystanie tego algorytmu zastosowano do rozpoznawania ręcznie pisanych cyfr [93]. W metodzie tej konstruowanych jest *M* modeli sieci SVM. Każdy z nich odpowiada za rozpoznanie jednej klasy. *I*-ta sieć uczona jest z użyciem danych treningowych, w których przykłady z *i*-tej klasy są kojarzone z $d_i = 1$, a pozostałe z $d_i = -1$. *I*-ta sieć rozwiązuje następujący problem:

$$\min_{w^i,b^i,\xi^i}\left[\frac{1}{2}(\mathbf{w}^i)^T\mathbf{w}^i + C\sum_{j=1}^{p}\xi_j^i\right],$$

$$(\mathbf{w}^i)^T\phi(\mathbf{x}_j)+b^i \geq 1-\xi_j^i \text{, jeśli } d_j = i, \tag{3.48}$$

$$(\mathbf{w}^i)^T\phi(\mathbf{x}_j)+b^i \leq -1+\xi_j^i \text{, jeśli } \; d_j \neq i,$$

$$\xi_j^i \geq 0 \text{, dla } \boldsymbol{j} = \mathbf{1}, \ldots, \boldsymbol{p},$$

gdzie dane uczące $\mathbf{x}_j$ są przenoszone do przestrzeni wielowymiarowej poprzez funkcję ϕ oraz parametr C.

Po rozwiązaniu (3.47) następuje etap odtwarzania, w którym ten sam wektor $\mathbf{x}$ podawany jest na wejście każdej sieci SVM. Określane są następnie wartości wszystkich funkcji decyzyjnych zgodnie z

$$\begin{aligned} y_1(\mathbf{x}) &= \mathbf{w}_1^T\boldsymbol{\varphi}\ \ \mathbf{x} \\ y_2(\mathbf{x}) &= \mathbf{w}_2^T\boldsymbol{\varphi}\ \ \mathbf{x} \\ &\ldots \\ y_M(\mathbf{x}) &= \mathbf{w}_M^T\boldsymbol{\varphi}\ \ \mathbf{x}\ . \end{aligned} \tag{3.49}$$

Decyzja o klasyfikacji do określonej klasy wektora $\mathbf{x}$ następuje poprzez wybór klasy o największej wartości funkcji decyzyjnej.

3.7.2 Algorytm typu „jeden przeciw jednemu”

Inną metodą jest algorytm *„jeden przeciw jednemu”* [94], w którym tworzonych jest $\frac{k(k-1)}{2}$ klasyfikatorów, a następnie każdy z nich trenowany jest na danych z dwóch różnych klas. Tę strategię w odniesieniu do SVM po raz pierwszy zastosował Friedman [95] oraz Kreßel [96]. Dla danych treningowych z i-tej iteracji i j-tej klasy, należy rozwiązać problem klasyfikacji dla dwóch klas:

$$\min_{w^{ij},b^{ij},\xi^{ij}}\left[\frac{1}{2}(\mathbf{w}^{ij})^T\mathbf{w}^{ij} + C\sum_t\xi_t^{ij}\right],$$

$$(\mathbf{w}^{ij})^T\phi(\mathbf{x}_t)+b^{ij} \geq 1-\xi_t^{ij} \text{, jeśli } d_t = i, \tag{3.50}$$

$$(\mathbf{w}^{ij})^T\phi(\mathbf{x}_t)+b^{ij} \leq -1+\xi_t^{ij} \text{, jeśli } \; d_t = j,$$

$$\xi_t^{ij} \geq 0\ .$$

3.7.3 Inne algorytmy wielokrotnej klasyfikacji

Innym algorytmem jest tzw. DAGSVM [97]. Etap uczenia jest taki sam jak w metodzie *„jeden przeciw jednemu”* poprzez rozwiązanie $\frac{k(k-1)}{2}$ binarnych sieci SVM. Inaczej jednak sieć się zachowuje na etapie testowania [87].

Istnieją także inne metody wielokrotnej klasyfikacji. Do jednej z nich można zakwalifikować np. metodę Crammera i Singera [98].

Przeprowadzone badania wykazały [87], że do praktycznych zastosowań najlepiej nadaje się metoda *„jeden przeciw jednemu”* oraz DAGSVM. W niniejszej pracy do korzystano z metody *„jeden przeciw jednemu”.*

3.8 Optymalizacja sieci SVM

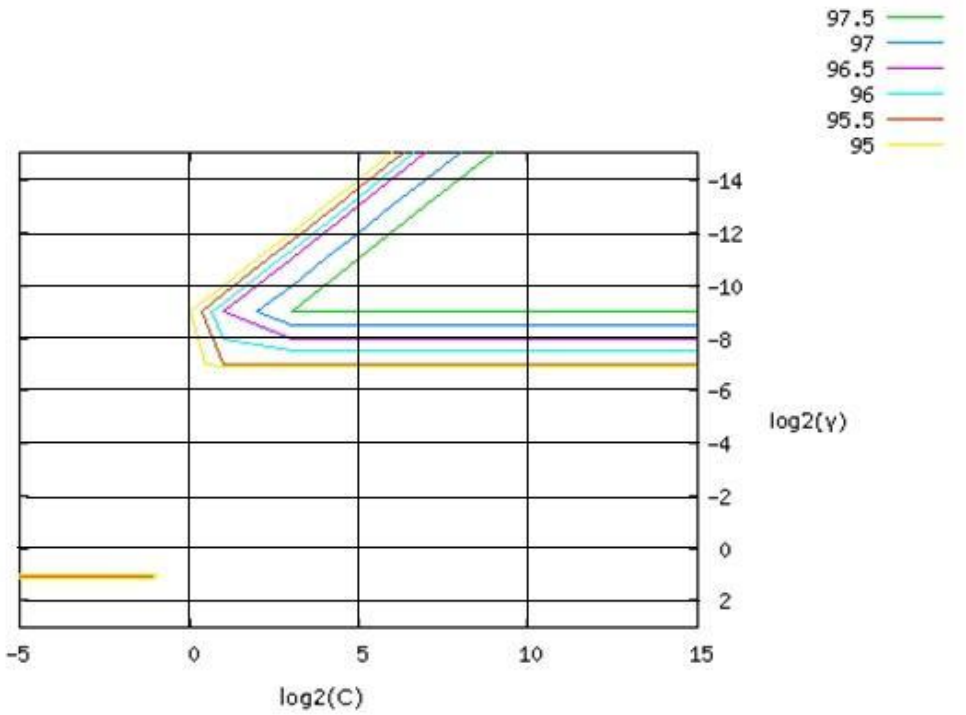
Aby tworzony model sieci SVM charakteryzował się jak największą skutecznością rozpoznawania, stosuje się kilka dodatkowych technik, które pokrótce zostaną przedstawione.

Normalizacja zakresu wartości danych w wektorze parametrów

Skalowanie danych jest bardzo istotne zarówno w tradycyjnych sieciach neuronowych jak i w sieciach typu SVM. Główną zaletą tego procesu jest to, że dzięki normalizacji zakresu atrybuty o większej wartości liczbowej nie dominują tak bardzo nad tymi o mniejszej wartości liczbowej. Kolejną zaletą jest przyspieszenie obliczeń. Ze względu na fakt, że wartości jądra zwykle zależą od wartości wektorów parametrów, duże wartości liczbowe mogą powodować problemy obliczeniowe w takich jądrach jak np. liniowe czy wielomianowe. Najczęściej stosowana jest normalizacja w zakresie od -1 do 1 lub od 0 do 1. W pracy na etapie klasyfikacji zdecydowano się na normalizację danych wejściowych w zakresie od -1 do 1.

Technika grid-search

W sieciach C-SVC wykorzystujących jądra radialne (patrz Tab. 3.2) najistotniejsze są dwa parametry: C i γ. Nie wiadomo z góry, jaka kombinacja tych dwóch parametrów jest optymalna do uzyskania najlepszego wyniku rozpoznawania. Ważne jest więc takie znalezienie pary parametrów C i γ, aby klasyfikator najlepiej rozpoznawał nieznane dane. W tym celu wykorzystuje się technikę zwaną w języku angielskim jako „*grid-search*”, co można próbować przetłumaczyć, jako wyszukiwanie na węzłach siatki. Polega ona na podziale obszaru wyszukiwania umowną siatką, a następnie wykorzystania do obliczeń danych dwóch parametrów, które podlegają zmianie w każdej iteracji. Jest to metoda prosta a co więcej, możliwe jest dokonywanie obliczeń dla kilku punktów w sposób równoległy. Wadą metody jest znaczny nakład obliczeniowy. Te same obliczenia należy powtórzyć tyle razy, ile jest węzłów siatki. Rys. 3.4 ilustruje działanie metody *grid-search*. W jego prawym górnym rogu zaznaczono kolorowe linie, którym przyporządkowana jest procentowa skuteczność uzyskana w czasie procesu uczenia sieci techniką *grid-search*.



Rys. 3.4 Ilustracja działania metody *grid-search*

Stosując metodę *grid-search* do uczenia sieci SVM, ostatecznie do modelu sieci wybierana jest taka para parametrów *C* i γ, dla której dokładność rozpoznawania jest najlepsza. Zbadano [91], że najlepszą drogą do znalezienia optymalnej pary parametrów *C* i γ jest ich zmiana metodą expotencjalnego wzrostu tych parametrów, np. $C=2^{-5}, 2^{-3},\ldots, 2^{15}$ oraz $\gamma=2^{-15}, 2^{-13}, \ldots, 2^{3}$.

Technika cross-validation

Należy zwrócić uwagę, że osiągnięcie wysokiej skuteczności rozpoznawania na danych treningowych może nie być reprezentatywne dla nowych danych testowych. Dlatego stosuje się rozdzielenie danych uczących na dwie grupy, z których jedna uważana jest za nieznaną dla klasyfikatora. Wtedy otrzymany wynik lepiej będzie odzwierciedlać skuteczność rozpoznawania nieznanych danych. Procedura ta nosi nazwę „*cross-validation*" [13][5], a jej twórcą był statystyk Seymour Geisser [13][6]. Metoda ta jest szczególnie przydatna, jeśli nie dysponuje się wystarczającą liczbą danych uczących. W tzw. technice „v*-fold cross-validation*" dane treningowe dzielone są na v podgrup o tej samej długości. Sekwencyjnie jedna podgrupa jest testowana w sieci uczonej na pozostałych $v-1$ podgrupach. Procedura ta może zapobiec problemowi nadmiernego dopasowania (ang. *overfitting*) [99].

W niniejszej pracy technikę *cross-validation* wykorzystuje się w połączeniu z metodą *grid-search* poszukiwania optymalnej pary C i γ, dlatego w dalszej części połączenie obu tych metod będzie zwane w skrócie CVGS (ang. *Cross-Validation Grid-Search*).

[5] hasło: cross-validation
[6] hasło: Seymour Geisser

4. Projekt i testy systemu rozpoznawania izolowanych słów z użyciem sieci typu SVM

Rozdział ten poświęcony jest projektowi i testom systemu rozpoznawania mowy z użyciem techniki wektorów podtrzymujących. W rozdziale opisano konkretne rozwiązania zastosowane w systemie oraz przeprowadzone doświadczenia. Na ich podstawie zaproponowano realizację systemu. Rozdział składa się z dwóch głównych części. Podrozdział 4.5 poświęcony został w całości algorytmowi wyboru początku i końca, natomiast w podrozdziale 4.6 badany jest wpływ poszczególnych elementów systemu na skuteczność rozpoznawania. Na podstawie wyników przeprowadzonych eksperymentów zaproponowano końcowe parametry projektowanego systemu rozpoznawania mowy.

4.1 Założenia projektowanego systemu rozpoznawania izolowanych słów

Projektując system rozpoznawania mowy przyjęto kilka istotnych założeń, które wcześniej ogólnie przedstawiono w podrozdz. 1.2. Po pierwsze system przeznaczony jest do rozpoznawania pojedynczych słów lub zwrotów. Nie projektowano go z myślą o rozpoznawaniu mowy ciągłej.

Założono także, że powinien on pracować w środowisku zaszumionym. Jest to założenie dość ogólne, ponieważ trudno jednoznacznie określić, czy system będzie pracował z zadaną skutecznością przy określonym SNR. Skuteczność działania systemu uzależniona jest od zbyt wielu zmiennych, m.in. właściwości osobniczych mówców, jakości nagrania, liczby klas i powtórzeń, słów wybranych jako klasy, itd. Skłoniło to autora do określenia możliwości pracy systemu przy określonym SNR dopiero na etapie wniosków z eksperymentów, w których zawarta jest już dokładna informacja o warunkach przeprowadzonych testów.

System przeznaczony jest głównie do pracy z jednym użytkownikiem, dlatego większość testów wykonywanych jest z użyciem bazy WBI (patrz podrozdz. 4.3.2). Możliwa jest także praca systemu, przy założeniu, że korzysta z niego wielu mówców a ich głosy zarówno są wykorzystywane na etapie uczenia jak i późniejszej normalnej pracy systemu.

Założono, że system będzie pracował z liczbą klas od kilkunastu do kilkudziesięciu. W większości badań system testowano z liczbą klas równą 20, choć w niektórych eksperymentach wykorzystywano mniejszą (np. 10 cyfr) lub większą liczbę klas (np. 33 litery alfabetu).

Aby uczynić system jak najbardziej użytecznym do pracy w realnym środowisku założono, że musi on zostać wyposażony w algorytm wyboru początku i końca wypowiedzi.

Założono także, że z systemu będzie można korzystać nagrywając mowę z użyciem komputera i zwykłego, niedrogiego mikrofonu. Założenie to jest spełnione dla wszystkich testów wykonywanych z użyciem bazy WBI, natomiast w przypadku bazy CORPORA (patrz 4.3.3) dodatkowo w sposób sztuczny wprowadzano zaszumienie.

4.2 Wyposażenie sprzętowe i programowe stanowiska laboratoryjnego

Wyposażenie sprzętowe stanowiska laboratoryjnego stanowił komputer wyposażony m.in. w:

- procesor: AMD Athlon 1,8 GHz,
- płytę główną: MSI K9NGM z kartą graficzną NVIDIA GeForce 6100,
- pamięć RAM: DDRII 2x512 MB (400MHz) w trybie Dual Channel,
- kartę dźwiękową: AC-97 (wbudowana na płycie)
- zestaw słuchawkowy z mikrofonem X-Power XP 6207M o parametrach mikrofonu: impedancja 2,2kΩ, czułość: -58dB +\- 3dB i paśmie od 20Hz do 20kHz.

Jak można zauważyć, zarówno karta dźwiękowa jak i zestaw słuchawkowy są stosunkowo niedrogimi, ogólnie dostępnymi narzędziami, co było jednym z założeń przy tworzeniu własnej bazy imion WBI.

Wyposażenie programowe składało się m.in. z:

- środowiska programistycznego Matlab [100]/Octave [101] wraz z pakietem wybranych programów narzędziowych,
- biblioteki do SVM [84],
- systemu operacyjnego Windows XP,
- oprogramowania Python 2.5 [102],
- programu do wykresów Gnuplot 4.0 [103].

Sygnały wejściowe stanowiły słowa, cyfry lub litery alfabetu, które pochodziły z trzech różnych baz danych:

- bazy nagrań sygnałów mowy polskiej CORPORA [104],
- własnej bazy imion WBI (patrz podrozdz. 4.3.2),
- własnej, pierwotnej bazy wypowiedzi [105].

Bibliotekę do sieci SVM, oprogramowanie Python, program Gnuplot a także własną bazę imion oraz użyte w pracy nagrania z własnej, pierwotnej bazy danych zostały zamieszczone na dołączonej do pracy płycie CD-ROM.

Szerszy opis każdej z baz danych wykorzystywanych w eksperymentach przedstawiono w kolejnym podrozdziale.

4.3 Bazy nagrań sygnałów mowy

W badaniach korzystano w sumie z trzech baz nagrań. W pracach nad algorytmem wyboru początku i końca słowa posłużono się starszą własną bazą wypowiedzi, nagraną pierwotnie na potrzeby pracy magisterskiej autora [105]. Baza danych CORPORA [104] jest dużo bardziej rozbudowana, ale wszystkie wypowiedzi mają już znaleziony początek i koniec słowa, więc utrudnione jest testowanie algorytmów ich znajdywania. Z kolei nowsza Własna Baza Imion nie była wykorzystywana w testach dotyczących algorytmu wyboru początku i końca sygnału

(patrz podrozdz. 4.5), ze względu na fakt, że powstała dopiero w późniejszym okresie. Dwie ostatnie bazy, czyli baza CORPORA i Własna Baza Imion WBI, były wykorzystywane w testach systemu z użyciem sieci SVM na etapie klasyfikacji oraz w końcowych badaniach skuteczności systemu.

Konieczność powstania bazy WBI wynikła z założeń pracy systemu. System ma pracować głównie z jednym mówcą. Zarówno starsza własna baza jak i baza CORPORA nie mają więcej niż maksymalnie trzy powtórzenia tej samej wypowiedzi przez jednego mówcę. Okazało się, że aby móc testować sieć na danych innych niż ją uczono, konieczna jest większa liczba nagrań każdej z klas.

4.3.1 Własna pierwotna baza wypowiedzi

Jak wspomniano wcześniej, wybrane wypowiedzi z pierwotnej bazy danych wykorzystywano wyłącznie do testów z podrozdz. 4.5, dotyczącym algorytmów wyboru początku i końca wypowiedzi.

Nagrania w bazie zawierają nagrane imiona, przezwiska, całe imiona wraz z nazwiskami lub nawet nazwy znanych mówcy miejsc. Cała baza nagrana jest przez jednego mówcę – autora pracy. Dobór słów w bazie mógłby np. symulować słownik używany w systemach wybierania połączeń głosem.

Mimo że baza zawiera znacznie więcej słów do testów wykorzystano następujące sygnały: *„dom”*, *„faraon”*, *„Justyna Kowalska”*, *„szkoła”* oraz *„Ola Mucha”*.

Nagrania zapisano w postaci cyfrowej z częstotliwością próbkowania 8kHz i 16-bitową rozdzielczością próbki sygnału.

4.3.2 Własna Baza Imion

Własna Baza Imion zwana dalej w skrócie WBI używana jest w drugiej grupie testów w podrozdz. 4.6. Przy tworzeniu bazy założono, że ma ona zostać nagrana z użyciem ogólnie dostępnego sprzętu do zapisywania mowy opisanego w podrozdz. 4.2. Mimo, że baza została nagrana z częstotliwością próbkowania 16kHz i 16-bitową rozdzielczością, to do testów korzystano z $F_p = 8kHz$, wykorzystując do zmiany częstotliwości próbkowania funkcję *resample* (patrz podrozdz. 4.6.6) z Matlaba. WBI zamieszczono w Dodatku B na płycie CD-ROM.

WBI stanowi 20 różnych imion nagranych przez jednego mówcę – autora pracy. Listę imion stanowią: *„Agata”, „Barbara”, „Czesław”, „Dominik”, „Ewa”, „Feliks”, „Grzegorz”, „Honorata”, „Iwona”, „Jarosław”, „Krystyna”, „Leszek”, „Małgorzata”, „Piotr”, „Radosław”, „Sławomir”, „Tomasz”, „Urszula”, „Wioletta”* i *„Zuzanna”*. Przy wyborze odpowiednich słów kierowano się założeniem, aby każde z nich rozpoczynało się od innej litery.

Nagranie każdego z imion powtórzono w sumie 20 razy, co daje w sumie 400 nagrań w bazie. Standardowo, 10 powtórzeń każdego ze słów wykorzystywane jest na etapie uczenia systemu, a pozostałe 10 wykorzystywane jest na etapie jego testowania. Wyjątkiem jest podrozdz. 4.6.12, gdzie badany jest wpływ liczby powtórzeń na skuteczność rozpoznawania.

4.3.3 Baza CORPORA

Z bazy danych CORPORA [104] autor mógł skorzystać dzięki uprzejmości prof. S. Grocholewskiego. Wszystkie nagrania, z których korzystano w badaniach, zostały nagrane

z użyciem mikrofonów pojemnościowych z częstotliwością próbkowania 16kHz i rozdzielczością 12 bitów. Do celów eksperymentów w większości testów zmniejszano częstotliwość próbkowania do 8KHz. Nagrań dokonano w warunkach naturalnych pomieszczeń, w bezpośredniej bliskości pracującego komputera, brak jednak informacji, jaki osiągnięto SNR.

Do eksperymentów wykorzystano następujące nagrania:

- 33 litery alfabetu fonetycznie: *„a, ą, be, ce, cie, de, e, ę, ef, ge, ha, i, jot, ka, el, eł, em, en, eń, o, pe, ku, er, es, eś, esz, te, u, wu, y, zet, ziet, żet”*,
- 10 cyfr – od *„zero”* do *„dziewięć”*,
- oraz 20 imion – takich samych jak we Własnej Bazie Imion.

Wypowiedzi zostały nagrane przez mężczyzn i kobiety (w tym dzieci) w wieku od 9 do 70 lat. Dzięki oznaczeniu każdego z mówców gwiazdką, możliwe jest rozróżnienie grupy wiekowej mówców zgodnie z oznaczeniami:

* 9 - 15 lat,
** 20 - 30 lat,
*** 30 - 50 lat,
**** 50 - 70 lat.

Doboru głosów mówców do większości badań dokonano zgodnie z kluczem zamieszczonym w artykule [106]. W testach z podrozdz. 4.6.4 i 4.6.14 dokonywano klasyfikacji spośród 20 imion, natomiast w przypadku eksperymentów z podrozdz. 4.6.15 każda z klas była jedną z 33 liter alfabetu.

Zbiór uczący tworzą nagrania następujących 21 osób (wg oznaczeń katalogów bazy CORPORA):

- głosy męskie: *ao1m1**, bc1m1**, jc1m1**, jk1m1**, jo1m1***, jp1m1***, jp2m1***, js1m1**, kd1m1****, kd2m1****, mr1m1**, ms1m1**, pl1m1, ps1m1****,
- głosy kobiece: *af1k1**, bc1k1**, bw1k1**, hk1k1***, is1k1***,
- głosy dziecięce: *ck1c1*, zk1d1**.

Zbiór testowy tworzą nagrania następujących 20 osób (wg oznaczeń katalogów):

- głosy męskie: *dg1m1**, pw1m1**, rg1m1****, sg1m1***, sg2m1***, sp1m1**, sw1m1**, ts1m1**, tz1m1**, wb1m1**, wm1m1**, zb1m1**, zk1m1*****,
- głosy żeńskie: *hk2k1***, ld1k1****, oj1k1**, pb1k1**, pl1k1*****,
- głosy dziecięce: *lk1d1*, zk2d1**.

Nagrania *hk1k1* i *zk1d1* ze zbioru uczącego, oraz *hk2k1* i *zk2d1* ze zbioru testowego pochodzą od tych samych osób, ale zostały zrealizowane w różnych okresach czasu.

Są jeszcze dwa testy, w których zastosowano bazę CORPORA. Tym razem rozpoznawanych jest 10 cyfr począwszy od *zera* do *dziesięć*. W tym wypadku zastosowano wszystkie dostępne nagrania cyfr z bazy CORPORA, korzystając wyjątkowo z tych samych danych na etapie uczenia jak i testowania sieci. W sumie wykorzystano 450 nagrań pochodzących od 45 mówców, które wykorzystano w podrozdz. 4.6.6 i 4.6.7. Przedział wiekowy mówców kształtuje się następująco: 6 mówców od 9-15 lat, 20 w wieku od 20 do 30 lat, 12 mówców od 30 do 50 lat oraz 7 mówców, którzy byli w wieku od 50 do 70 lat.

4.4 Opis testów w środowisku zaszumionym

4.4.1 Stosunek sygnału do szumu

Poziom sygnału mowy do szumu określa stosunek sygnału do szumu SNR (ang. *Signal to Noise Ratio*) [13][7], definiowany jako:

$$SNR = 10\log(\frac{P\mathbf{x}}{P\mathbf{y}}) = 20\log(\frac{\mathbf{x}_{RMS}}{\mathbf{y}_{RMS}})[dB], \tag{4.1}$$

gdzie:

$\boldsymbol{P}_{\mathbf{x}}$ – średnia energia sygnału mowy bez szumu,

$\boldsymbol{P}_{\mathbf{y}}$ – średnia energia szumu,

$\mathbf{x}_{\boldsymbol{RMS}}$ – wartość skuteczna RMS (ang. *Root Mean Square*) sygnału mowy bez szumu,

$\mathbf{y}_{\boldsymbol{RMS}}$ – wartość skuteczna szumu.

Ze względu na fakt, że baza CORPORA zawiera w większości nagrania zawierające jedynie sygnał mowy założono, że poziom sygnału mowy można określić wyliczając wartość skuteczną [13][8] całego sygnału $\mathbf{y}_{\boldsymbol{RMS}}$ zgodnie z

$$\mathbf{x}_{RMS} = \frac{\|\mathbf{x}\|}{\sqrt{N_x}} = \sqrt{\frac{1}{N_x}\sum_{i=1}^{N_x} x_i^{\,2}}\,, \tag{4.2}$$

gdzie:

$\|\mathbf{x}\|$ – oznaczenie normy wektora zdefiniowanej w (3.5),

$\mathbf{x}$ – sygnał mowy bez szumu,

$\boldsymbol{N_x}$ – liczba próbek sygnału mowy.

Podobnie postąpiono definiując wartość skuteczną szumu $\mathbf{y}_{\boldsymbol{RMS}}$ jako

$$\mathbf{y}_{RMS} = \frac{\|\mathbf{y}\|}{\sqrt{N_y}} = \sqrt{\frac{1}{N_y}\sum_{i=1}^{N_y} y_i^{\,2}}\,, \tag{4.3}$$

gdzie:

[7] hasło: Signal-to-noise ratio
[8] hasło: Root mean square

$\mathbf{y}$ – szum,

$\boldsymbol{N_y}$ – liczba próbek szumu.

Długość wektora szumu uzależniona jest od rodzaju przeprowadzonego testu. Dla testów skuteczności rozpoznawania z wyznaczonym a priori początkiem i końcem sygnału bez stosowania algorytmu wyboru początku i końca słowa, liczba próbek szumu była równa liczbie próbek sygnału.

Gdy dodatkowo testowano skuteczność działania różnych wersji algorytmów VAD z użyciem bazy CORPORA, sygnał szumu poszerzono o dodatkowe 300ms, dodając po 150ms przed oryginalnym sygnałem i taką samą długość za sygnałem. Ma to na celu urealnienie procesu rozpoznawania. Z reguły przed i po zakończeniu wypowiedzi następuje cisza. W przypadku środowiska zaszumionego w tym czasie obecny jest więc tylko szum, z którego pobierane są informacje o środowisku zaszumionym (np. w pierwszych czy ostatnich 100ms nagrania).

Najpierw obliczano wartość skuteczną całego sygnału mowy zgodnie z (4.2). Następnie, znając wartość zakładanego SNR, obliczano wartość skuteczną szumu zgodnie z zależnością

$$\mathbf{y}_{RMS} = \frac{\mathbf{x}_{RMS}}{10^{SNR/20}} . \tag{4.4}$$

Na koniec generowano szum różowy o odpowiednim poziomie $\mathbf{y}_{\boldsymbol{RMS}}$.

Sytuacja wygląda nieco inaczej przy wykonywaniu testów systemu, z wykorzystaniem algorytmu VAD na Własnej Bazie Imion. Sygnały tak zostały nagrane, aby algorytm samodzielnie znalazł początek i koniec sygnału, więc każdy z sygnałów zawiera zarówno sygnał mowy jak i sygnał zaszumiony. Zbadano na kilku przykładach, że średni poziom SNR bazy WBI wynosi ok. 32dB. W związku z tym nie mają sensu testy sytemu przy wyższym stosunku sygnału do szumu.

Należy zauważyć, że w przypadku badań z wykorzystaniem bazy WBI pojawia się problem z określeniem rzeczywistego poziomu sygnału do szumu, ponieważ brak w tym przypadku a priori takiej informacji. Założono więc, że cały sygnał będzie potraktowany jako sygnał mowy do wyznaczenia poziomu $\mathbf{x}_{\boldsymbol{RMS}}$. Na jego podstawie zostaje obliczony poziom $\mathbf{y}_{\boldsymbol{RMS}}$ zgodnie z (4.4), a następnie generowany jest szum, który jest dodawany do całego sygnału. Uproszczenie takie powoduje, że zaniżana jest wartość $\mathbf{x}_{\boldsymbol{RMS}}$ a przez to także $\mathbf{y}_{\boldsymbol{RMS}}$. Błąd jest tym większy, im więcej w nagranym sygnale stanowi jedynie szum bez mowy. Dodatkowo dodawanie szumu do sygnału nie uwzględnia już obecnego w sygnale szumu związanego z nagrywaniem. Oba te uproszczenia powodują, że dany poziom SNR może różnić się od jego rzeczywistej wartości dla każdego z sygnałów z Własnej Bazy Imion.

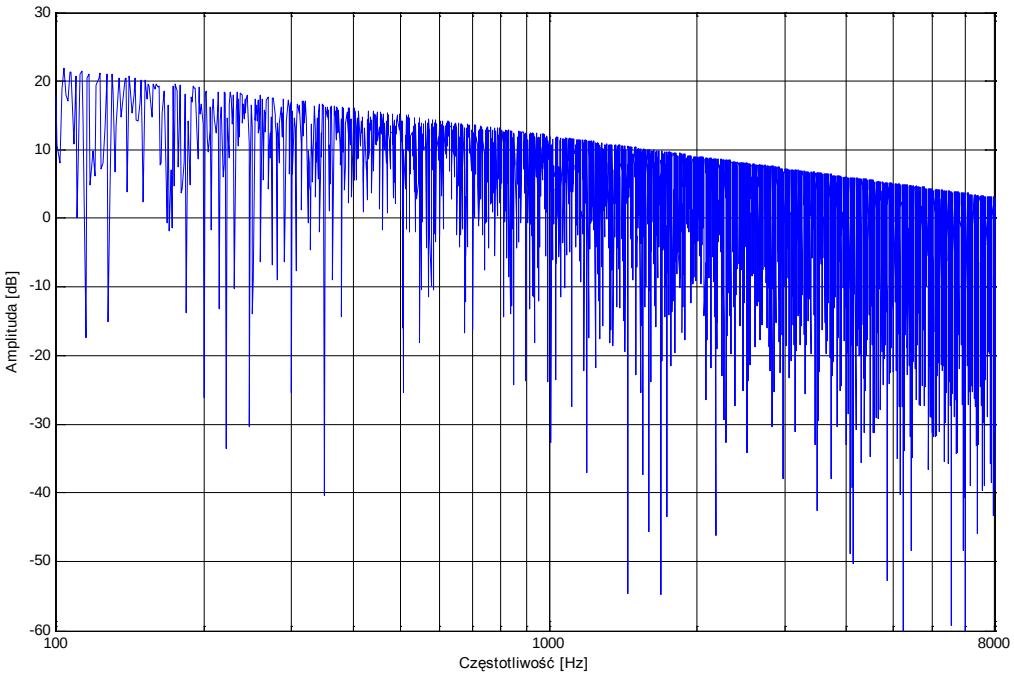
Należy także wspomnieć, że w przypadku bazy CORPORA, aby zbadać skuteczność działania projektowanego systemu z algorytmem wyboru początku i końca, poddano go testom, dodając do oryginalnego sygnału szum o takim natężeniu, aby SNR = 45dB. Proces zaszumienia sygnału przeprowadzono w sposób sztuczny, dodając szum różowy (patrz podrozdz 4.4.2) do oryginalnego

sygnału przed przystąpieniem do jakiejkolwiek dalszej obróbki sygnału. Wszystkie operacje były więc wykonywane na sygnale zaszumionym.

4.4.2 Wybór typu szumów do testów

Decydując o wyborze szumu akustycznego do testów systemu w środowisku zaszumionym kierowano się tym, aby szum odzwierciedlał wpływ rzeczywistych zakłóceń. Ze względu na fakt, że każde środowisko zewnętrzne charakteryzuje się inną charakterystyką szumu, nie ma idealnego źródła, które może reprezentować każde środowisko.

W badaniach nad sygnałami akustycznymi bardzo często jako wzorzec testowy zakłóceń wybierany jest tzw. szum różowy (ang. *pink noise, flicker noise)* [13][9], [107][10], którego charakterystykę w dziedzinie częstotliwości przedstawiono na Rys. 4.1. Ze względu na fakt, że jest on sygnałem, którego widmo częstotliwościowe oraz widmowa gęstość mocy są proporcjonalne do odwrotności częstotliwości, nazywany jest także szumem $\mathbf{1}/\boldsymbol{f}$ (ang. *one over f noise).* Widmowa gęstość mocy szumu różowego opada 10dB na dekadę czyli ok. 3dB na oktawę.



Rys. 4.1 Szum różowy w dziedzinie częstotliwości

W rzeczywistości wygenerowanie prawdziwego szumu różowego jest niemożliwe, ponieważ energia takiego sygnału byłaby nieskończona. Oznaczałoby to, że energia szumu różowego w każdym przedziale częstotliwości od $\boldsymbol{f_1}$ do $\boldsymbol{f_2}$ jest proporcjonalna do $\boldsymbol{log(f2\,/\,f1)}$, a jeśli $\boldsymbol{f_2}$ jest nieskończona to energia także. W praktyce więc szum różowy jest rzeczywiście „różowy" jedynie w ograniczonym zakresie częstotliwości, w paśmie przenoszenia karty dźwiękowej.

[9] hasło: Pink noise
[10] hasło: szum różowy

Nasuwa się pytanie, dlaczego nie wykorzystano szumu białego [107][11] do badań. Ze względu na fakt, że ludzkie ucho odbiera bodźce w skali logarytmicznej a nie liniowej, najbardziej słyszalne są wysokie tony w przypadku szumu białego. Wynika to z faktu, że większość odbieranych bodźców skupiona jest w najwyższej słyszalnej oktawie. W zakresie od 10kHz do 20kHz skupione jest tysiąc razy więcej mocy, niż w oktawie 10Hz do 20Hz. Podsumowując, szum różowy dużo lepiej reprezentuje zakłócenia odbierane przez ucho ludzkie niż szum biały, dlatego też ten ostatni nie został wykorzystany w niniejszej pracy.

Uwaga: Oznaczenie „brak szumu" lub „brak zewnętrznego szumu" (lub „szum naturalny") oznacza, że nie dokładano w sposób sztuczny szumu do sygnału. Nie oznacza to jednak, że on nie występuje. Jest to naturalny szum powstały w czasie nagrywania w warunkach domowych w bezpośredniej bliskości komputera. SNR dla poszczególnych sygnałów między całym sygnałem a pierwszymi 100ms wynosi średnio ok. 40dB w przypadku Własnej Pierwotnej Bazy Danych.

4.5 Opracowanie i testy nowego algorytmu wyboru początku i końca wypowiedzi z użyciem sieci typu One Class SVM.

Algorytm wyboru początku i końca sygnału jest jednym z kluczowych elementów systemu rozpoznawania izolowanych słów. Jest on niezbędny, aby w sposób prawidłowy przebiegł dalszy proces parametryzacji i klasyfikacji sygnału mowy.

W podrozdz. 4.5.1 przedstawiono opis zmodyfikowanego algorytmu Rabinera-Sambura wyboru początku i końca wypowiedzi, zwanego dalej w skrócie MRSED (ang. *Modified Rabiner-Sambur Endpoint Detection Algorithm*) [108]. Następnie w podrozdz. 4.5.2 wykonano wstępne testy skuteczności algorytmu zarówno w środowisku niezaszumionym, jak i zaszumionym przy SNR = 20dB, 12dB i 4dB.

Na podstawie wyników badań autor postanowił szukać nowych rozwiązań, które poprawiłyby skuteczność działania algorytmu VAD w środowisku zaszumionym. I tak w podrozdz. 4.5.3 przedstawiono parametry, które wykorzystano w rekomendacji ITU-T G729B. Przeprowadzono także dodatkowe badania mające na celu zbadanie, jaki wpływ ma zastosowanie filtru preemfazy na liczbę przejść przez zero (patrz podrozdz. 4.5.4), a także zastanawiano się nad zmianą definicji, zgodnie z którą liczona jest energia sygnału. Zbadano także w podrozdz. 4.5.5 wpływ na reprezentację sygnału operacji podnoszenia do kwadratu parametrów ΔZCR. Ponadto w podrozdz. 4.5.6 rozważono możliwość zastosowania badanych parametrów na dalszym etapie parametryzacji sygnału mowy.

Następnie w podrozdz. 4.5.7 zaimplementowano wybrane parametry do całkowicie nowego algorytmu wyboru początku i końca sygnału z użyciem specjalnego typu sieci One Class SVM. Na koniec porównano algorytmy MRSED i VAD-1SVM w podrozdz. 4.5.8 oraz podsumowano uzyskane wyniki w podrozdz. 4.5.9.

[11] hasło: biały szum

4.5.1 Zmodyfikowany algorytm L. R. Rabinera i M. R. Sambura (MRSED)[12]

Prace badawcze rozpoczęto od próby zaadoptowania zmodyfikowanego przez autora [108] algorytmu wyboru początku i końca słowa opartego na algorytmie L. R. Rabinera i M. R. Sambura [16]. Zmodyfikowany algorytm, nazywany dalej w skrócie algorytmem MRSED (ang. *Modified Rabiner-Sambur Endpoint Detection algorithm*), w odróżnieniu od oryginalnego rozwiązania [16], opiera się wyłącznie na pomiarze energii sygnału, jednocześnie pomijając drugi parametr, czyli liczbę przejść przez zero (ang. *Zero Crossing Rate - ZCR*). Zmodyfikowano w nim ponadto progi do wyznaczania końca wypowiedzi oraz wprowadzono kilka innych mniejszych zmian szczegółowo opisanych w [108].

Podobnie jak w oryginalnym algorytmie, sygnał po próbkowaniu i filtracji dzielony jest na ramki o długości 10ms. W obu algorytmach zakłada się, że pierwsze 100ms nagrywanego sygnału nie jest sygnałem mowy, a jedynie szumem tła. Z tego okresu czasu pobierane są informacje o średniej energii, a w oryginalnym algorytmie dodatkowo o średniej liczbie przejść przez zero szumu tła.

Energia m-tej ramki sygnału s w algorytmie MRSED liczona jest według (2.4), a w oryginalnym algorytmie zgodnie z zależnością

$$E_s[m] = \sum_{n=(m-1)N}^{mN-1} \left| s[n] \right|, \tag{4.5}$$

gdzie:

N – liczba próbek w ramce (np. dla okna długości 10ms $N = 100$ przy $Fs = 8kHz$),

m – numer ramki,

n – numer próbki sygnału.

Różnica polega na tym, że w proponowanej modyfikacji autor podnosi próbki do kwadratu, celem uwypuklenia próbek o większej wartości i zmniejszenia tych o mniejszej wartości. W prosty sposób jest tu niejako przeprowadzana dodatkowo operacja odszumiania.

Następnie w obu algorytmach obliczana jest zgodnie z (2.4) maksymalna energia sygnału IMX oraz energia szumu IMN pierwszych 100ms. Na ich podstawie liczone są następnie parametry $I1$ i $I2$, zgodnie z

$$I1 = 0.03(IMX - IMN) + IMN \tag{4.6}$$

oraz

[12] Podrozdział 4.5.1 opracowano głównie na podstawie własnej pracy magisterskiej [105] i publikacji autora [108].

$$I2 = 4IMN\,. \tag{4.7}$$

Ze wzoru (4.7) widać, że $I1$ jest w przybliżeniu wartością odpowiadająca 3% wartości szczytowej energii IMX. Z kolei $I2$ zgodnie z (4.8) odpowiada czterokrotnej wartości energii szumu tła IMN.

Następnie z parametrów $I1$ i $I2$ wybierany jest ten o mniejszej wartości, który następnie definiuje niższy próg ITL według wzoru

$$ITL = \min(I1, I2)\,. \tag{4.8}$$

Górny próg ITU obliczany jest na podstawie znajomości niższego progu ITL zgodnie z

$$ITU = 5ITL\,. \tag{4.9}$$

Dodatkowo w oryginalnym algorytmie [16], z czego zrezygnowano w zmodyfikowanej wersji algorytmu [108], obliczany jest próg liczby przejść przez zero dla szumu tła $IZCT$ (ang. ***s**Ilence **Z**ero **C**rossing **T**hreshold*) zgodnie z

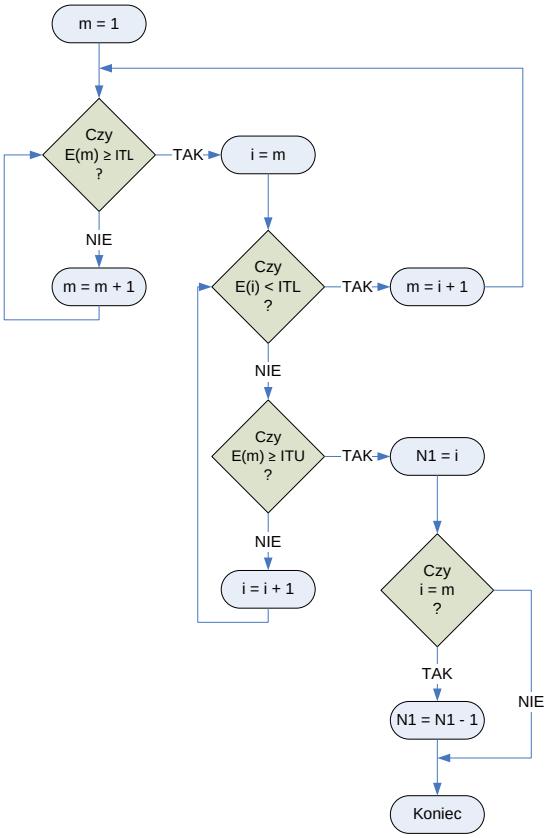
$$IZCT = \min(IF, IZC + 2\delta_{IZC})\,, \tag{4.10}$$

gdzie:

IF – 25 przejść przez zero w ciągu 10ms,

IZC – średnia zmierzona liczba przejść przez zero dla szumu tła mierzona w ciągu pierwszych 100ms sygnału,

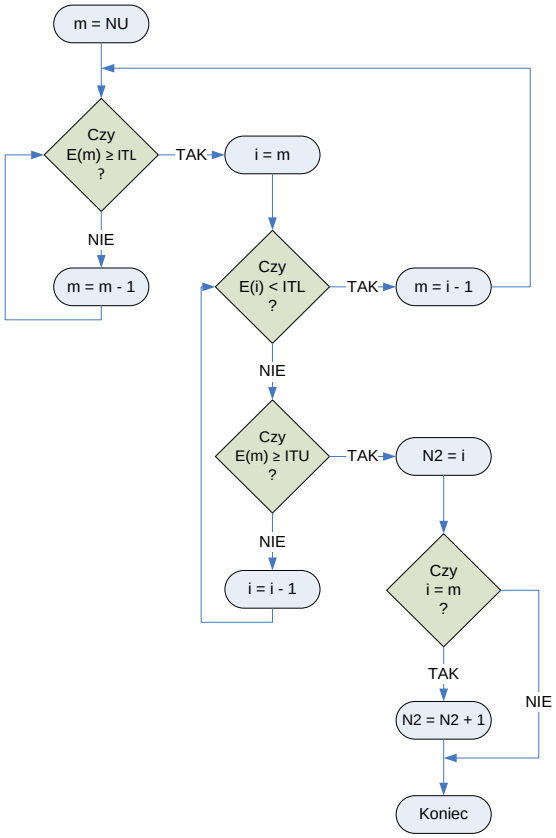
δ_{IZC} – odchylenie standardowe liczby przejść przez zero.



Rys. 4.2 Algorytm wyznaczania początku słowa wykorzystujący pomiar energii sygnału

Schemat blokowy działania algorytmu MRSED do znajdowania początku słowa, opartego wyłącznie na energii sygnału przedstawiono na Rys. 4.2. Algorytm początkowo sprawdza, dla której ramki zostanie przekroczony dolny próg ITL. Ramka ta, wstępnie uważana jest za początek słowa do momentu, w którym poziom energii znów spadnie poniżej ITL przed wzrostem ponad górny próg ITU. W takim wypadku ponownie wyszukiwana jest ramka, w której energia wzrasta powyżej ITL, a następnie wzrasta powyżej ITU. Ramka, w której energia przekroczyła ITU, oznaczana jest przez N_1.

Podobny algorytm, tym razem do znajdywania końca słowa, przedstawiony został na Rys. 4.3. Różnica polega na tym, że algorytm ten rozpoczyna działanie od końca sygnału, oznaczanego jako NU.

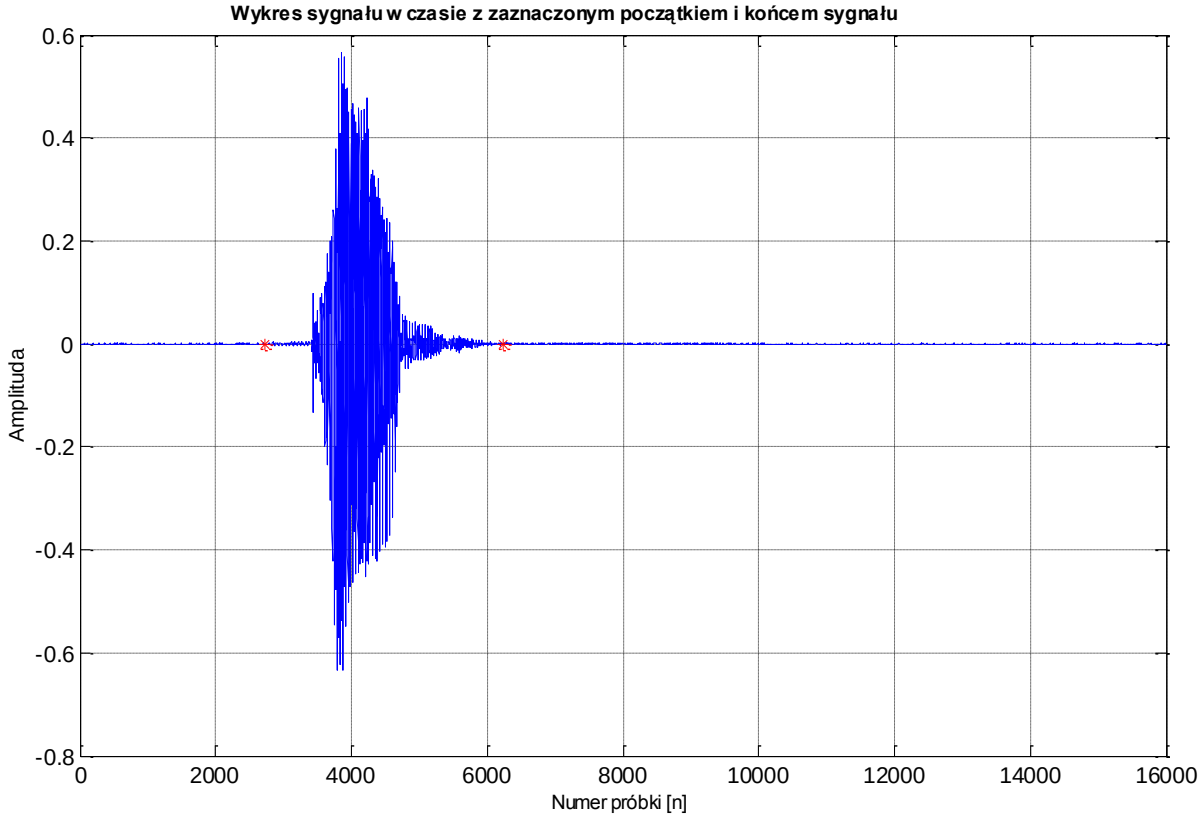


Rys. 4.3 Algorytm wyznaczania końca słowa wykorzystujący pomiar energii sygnału

Autor zastanawiał się pierwotnie nad możliwościami wprowadzenia zmian w algorytmie wyboru końca wypowiedzi po stwierdzeniu, że w niektórych przypadkach analizy wyboru końca słowa, zgodnie z oryginalnym algorytmem Rabinera-Sambura, punkt końca leżał za obszarem sygnału mowy tam, gdzie był rejestrowany szum. W celu zwiększenia skuteczności algorytmu autor zaproponował dodatkowo liczenie progów ITL i ITU z wykorzystaniem średniej energii liczonej w ostatnich 100ms nagrywanego sygnału, co spowodowało lepszą adaptację do poziomu szumu końcowej części rejestrowanego sygnału.

Uzupełnienie algorytmu znajdywania początku i końca słowa, bazującego na pomiarze energii sygnału, o algorytm liczby przejść przez zero powinno zwiększyć dokładność wyboru początku i końca słowa. Nie stwierdzono jednak w sposób jednoznaczny, że połączenie takie jest niezbędne, dlatego zrezygnowano z tej opcji w algorytmie MRSED [108]. W ostatnim czasie autor natrafił także na inne badanie potwierdzające słuszność tej teorii [109].

Przykład działania algorytmu MRSED z zaznaczonym początkiem i końcem słowa bazującym jedynie na pomiarze energii sygnału przedstawia Rys. 4.4.



Rys. 4.4 Wykres sygnału słowa *„dom”* z zaznaczonymi punktami początku i końca słowa w przypadku braku zewnętrznego szumu

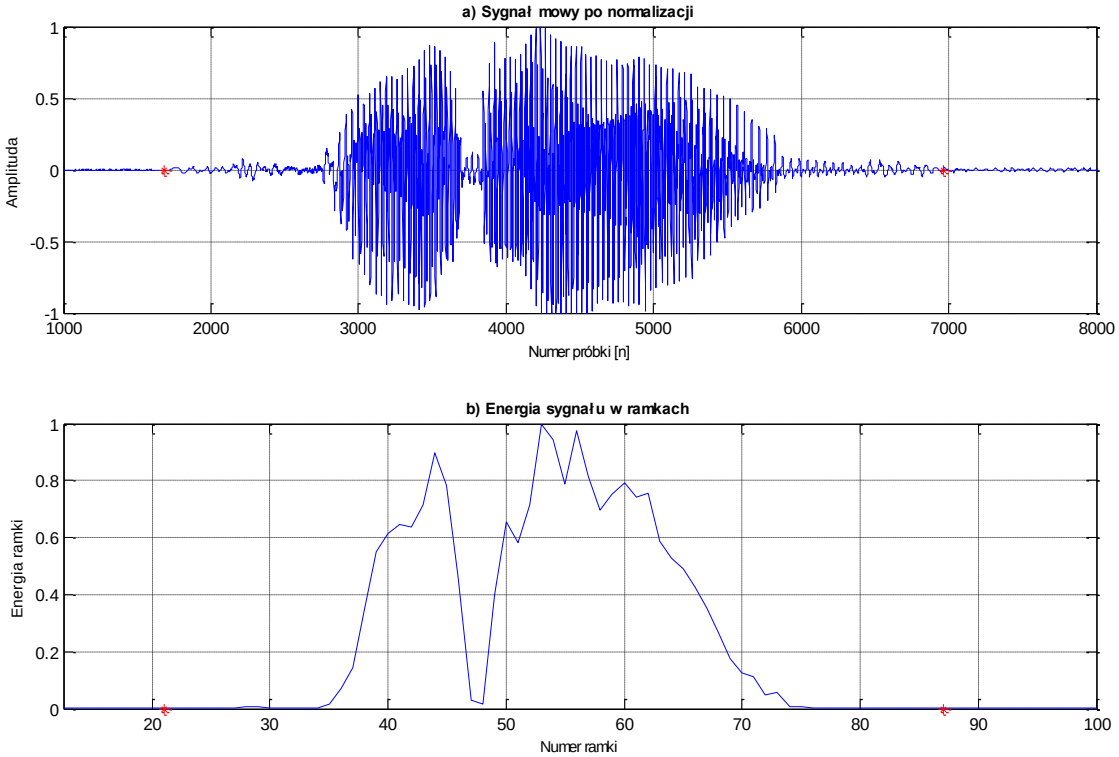
Należy zaznaczyć, że proponowana modyfikacja algorytmu wyboru początku i końca słowa Rabinera-Sambura nie była pierwotnie projektowana z myślą o pracy w środowisku zaszumionym. W celu sprawdzenia skuteczności algorytmu w obecności szumu, poddano go testom, które opisano w podrozdz. 4.5.2.

4.5.2 Wstępne testy algorytmu MRSED w środowisku zaszumionym oraz wpływ filtru preemfazy na skuteczność działania algorytmu

Głównym celem badania jest zbadanie skuteczności zmodyfikowanego algorytmu Rabinera-Sambura (algorytmu MRSED) w środowisku zaszumionym. Drugim celem tego badania jest zbadanie wpływu zastosowania filtru preemfazy na skuteczność działania algorytmu.

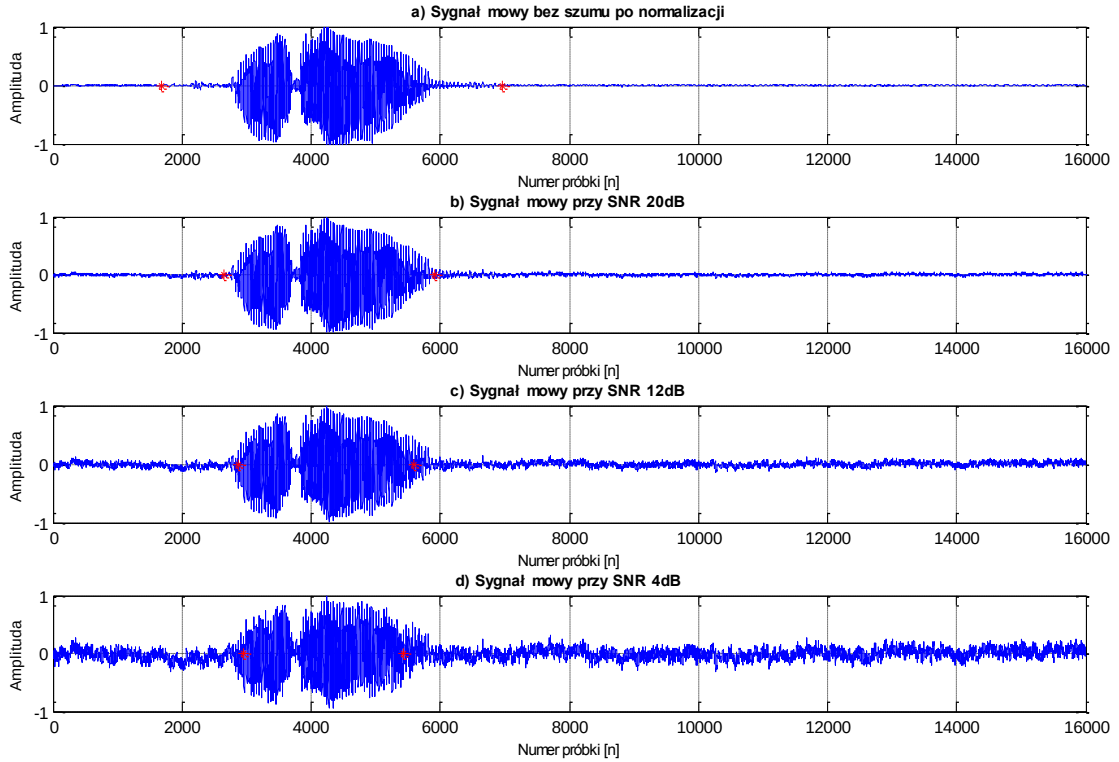
Do badań zastosowano następujące wypowiedzi z własnej pierwotnej bazy danych: *faraon*, *Justyna Kowalska* i *szkoła*.

Wykonano badania porównawcze przy naturalnym SNR, (czyli w przypadku braku zewnętrznego szumu), przy $SNR = 20dB$ i $12dB$ oraz w przypadku sygnału bardzo zaszumionego przy $SNR = 4dB$. Wszystkie testy w podrozdziałach od 4.5.2 do 4.5.8 wykonano przy częstotliwości próbkowania 8kHz.

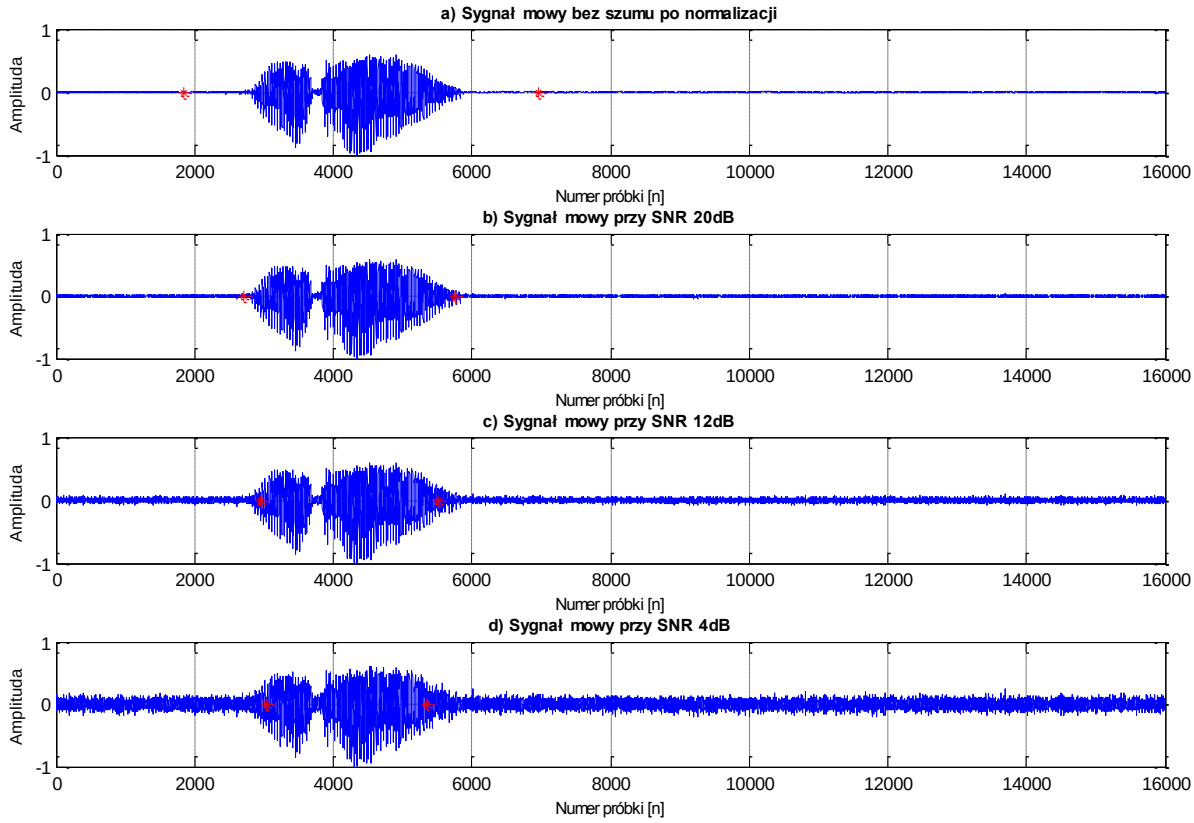


Rys. 4.5 Przebieg czasowy (a) i energia w poszczególnych ramkach (b) słowa *„faraon”* z zaznaczonym początkiem i końcem wypowiedzi (brak zewnętrznego szumu, zastosowano filtr preemfazy)

Jak widać na Rys. 4.5 algorytm MRSED stosunkowo dobrze sobie radzi z wyborem początku i końca w przypadku braku zewnętrznego szumu po zastosowaniu filtru preemfazy.



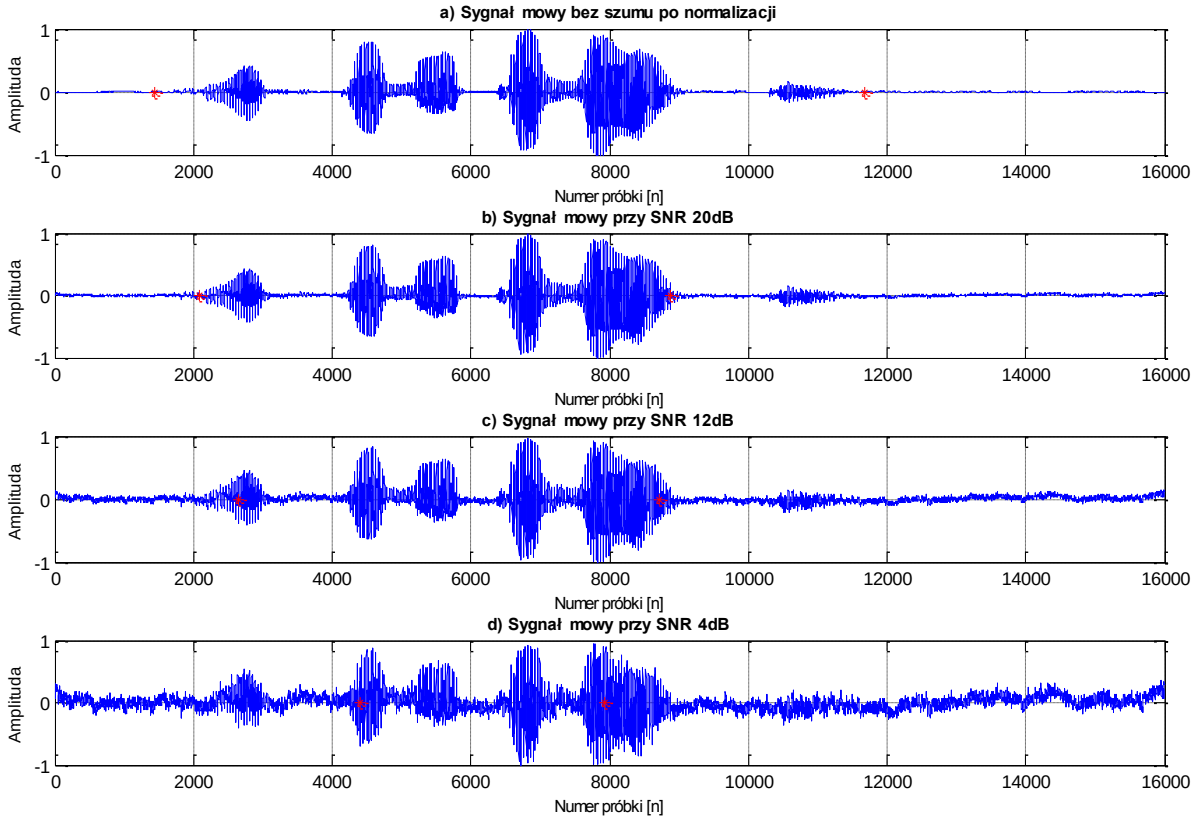
Rys. 4.6 Wybór początku i końca sygnału dla wypowiedzi "*faraon*" przy różnym SNR (nie zastosowano filtru preemfazy)



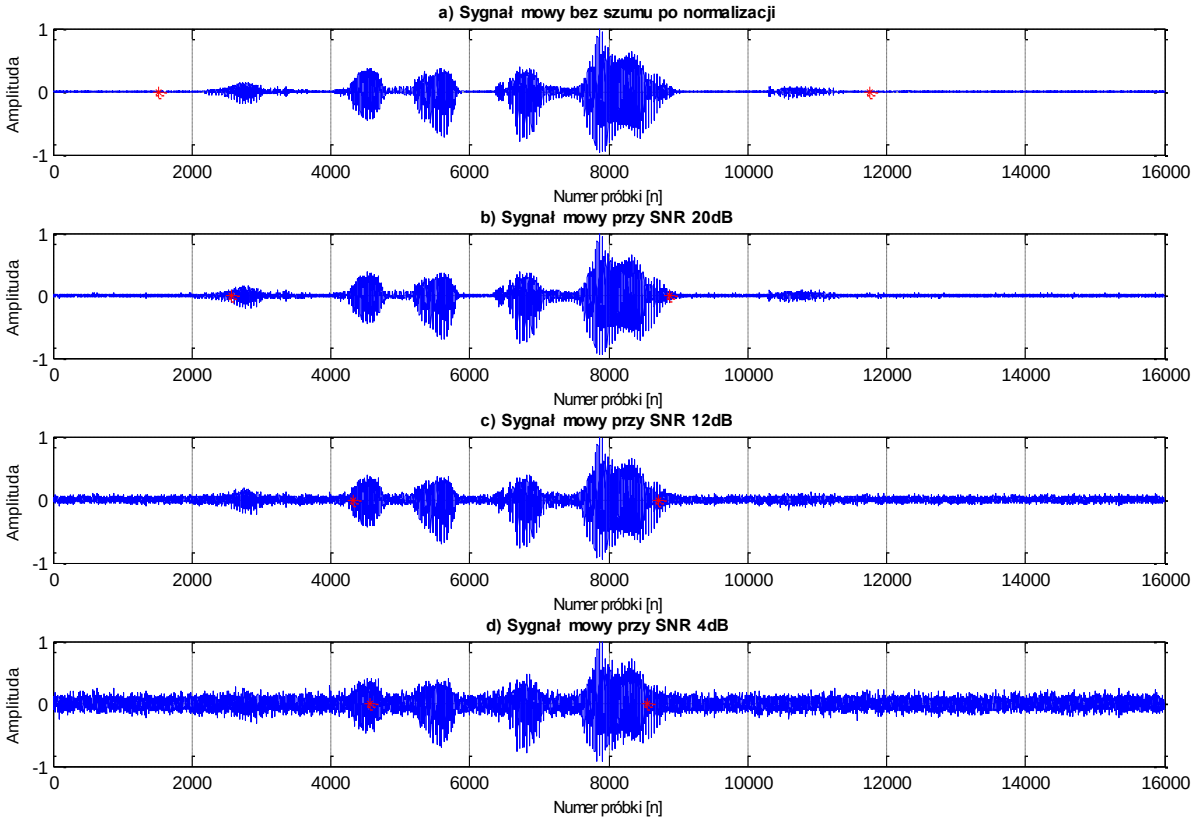
Rys. 4.7 Wybór początku i końca sygnału dla wypowiedzi „*faraon*" przy różnym SNR (po przejściu przez filtr preemfazy)

Na Rys. 4.6 i Rys. 4.7 porównywano sygnał słowa „*faraon*" przy różnym SNR. Zastosowanie filtru preemfazy praktycznie nie miało wpływu na skuteczność działania algorytmu. Można jednak wyraźnie zaobserwować pewne „ustabilizowanie się" sygnału zaszumionego. Zjawisko to tym bardziej jest widoczne, im niższy jest SNR, co najlepiej można zaobserwować na przebiegach na Rys. 4.6d) i Rys. 4.7d) przy SNR = 4dB.

Im większy jest szum, tym bardziej wyniki rozpoznawania początku i końca słowa pogarszają się. Algorytm coraz bardziej zawęża obszar, który rozpoznaje jako ten, który zawiera mowę.

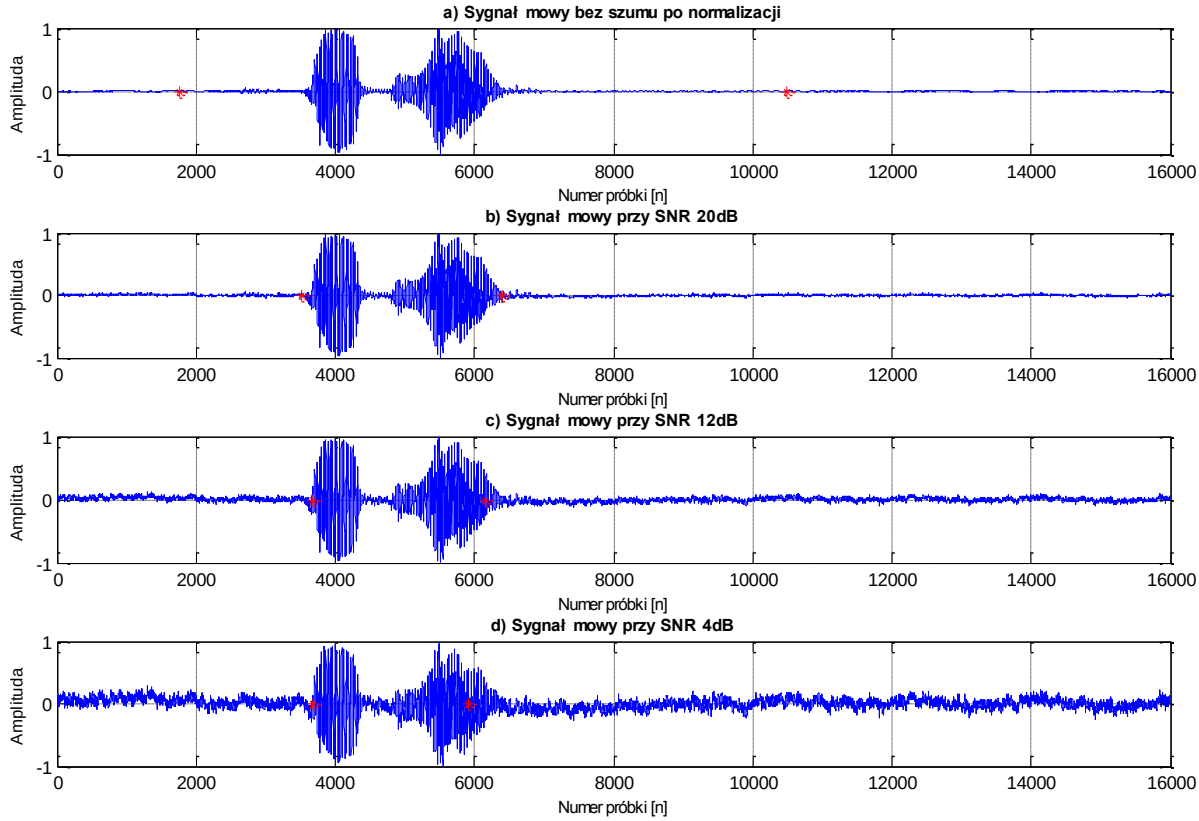


Rys. 4.8 Wybór początku i końca sygnału dla wypowiedzi „*Justyna Kowalska*" przy różnym SNR (nie zastosowano filtru preemfazy)

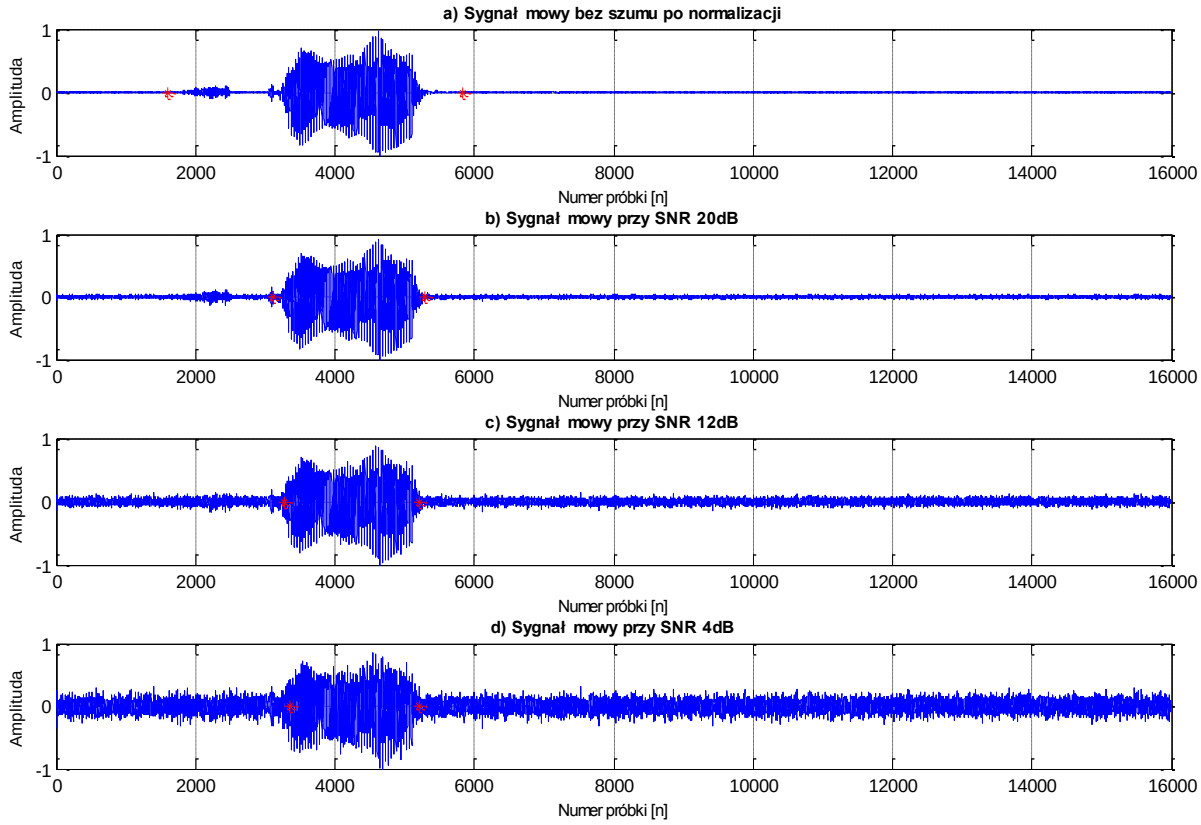


Rys. 4.9 Wybór początku i końca sygnału dla wypowiedzi „*Justyna Kowalska*" przy różnym SNR (po przejściu przez filtr preemfazy)

W przypadku wypowiedzi *„Justyna Kowalska”* algorytm stosunkowo dobrze działa w przypadku braku zewnętrznego szumu. Wpływ zastosowania preemfazy nie wpływa znacząco na wyniki działania algorytmu, widać jednak pewną różnicę. Na Rys. 4.8c) i Rys. 4.9c), w przypadku, w którym zastosowano filtr preemfazy, wyniki rozpoznawania początku słowa pogorszyły się, pomijając pierwszą część wypowiedzi. Autor uważa, że jest to spowodowane nie tyle negatywnym wpływem działania filtru preemfazy, co losowym generowaniem szumu osobno dla każdego z badanych przypadków.



Rys. 4.10 Wybór początku i końca sygnału dla wypowiedzi *„szkoła”* przy różnym SNR (nie zastosowano filtru preemfazy)



Rys. 4.11 Wybór początku i końca sygnału dla wypowiedzi *„szkoła”* przy różnym SNR (po przejściu przez filtr preemfazy)

Dla słowa *„szkoła”* wyniki na Rys. 4.10 i Rys. 4.11 są także porównywalne oprócz sytuacji braku zewnętrznego szumu. Widać wyraźnie na Rys. 4.10a), że brak filtru preemfazy mógł spowodować w tym przypadku błędną klasyfikację zarówno początku, jak i końca sygnału. Oba punkty zostały zbyt wcześnie wyznaczone, co świadczy, że w tym przypadku poziomy decyzyjne ITU i ITL były ustawione na zbyt niskim poziomie.

Podsumowując wyniki przeprowadzonych badań, które przedstawiono od Rys. 4.6 do Rys. 4.11, dotyczących skuteczności zmodyfikowanego algorytmu Rabinera-Sambura do pracy w środowisku zaszumionym należy stwierdzić, że słabą stroną algorytmu jest problem w trafnej lokalizacji tych części wypowiedzi, które zawierają głoski bezdźwięczne np. *„s”*, *„f”*, itp. Ponadto w przypadku wzrostu zaszumienia zwiększa się także energia szumu, co prowadzi do pogorszenia jakości rozpoznawania.

Można powiedzieć, że już w przypadku szumu na poziomie SNR = 20dB algorytm ma tendencję do identyfikowania zarówno początku jak i końca wypowiedzi błędnie, powodując pomijanie części sygnału i interpretowanie go jako szum.

W czasie testów badano także wpływ filtru preemfazy na działanie algorytmu. Na podstawie dotychczasowych badań nie można jednoznacznie stwierdzić, czy preemfaza poprawia lub pogarsza działanie prezentowanego algorytmu VAD.

W związku z nieuzyskaniem zadowalających wyników skuteczności algorytmu MRSED, autor postanowił rozpocząć prace nad algorytmem o wyższej skuteczności w czasie pracy w środowisku zaszumionym.

4.5.3 Zastosowanie parametrów z rekomendacji ITU-T G729B

Jak pokazano w podrozdz. 4.5.2, algorytm wyboru początku i końca słowa, oparty jedynie na pomiarze energii sygnału, może nie znajdywać skutecznie rzeczywistego początku i końca słowa w środowisku zaszumionym.

Autor, poszukując skuteczniejszych rozwiązań, zainteresował się parametrami stosowanymi w rekomendacji Międzynarodowej Unii Telekomunikacyjnej (ang. *International Telecommunication Union - ITU*) oznaczonej jako G.729B [110].

W rekomendacji tej stosowana jest jednoczesna kombinacja kilku parametrów. Należą do nich: parametry LSF (ang. *Line Spectral Frequencies*), dwa typy energii, a także liczba przejść przez zero.

Parametry LSF zostały szczegółowo zdefiniowane w podrozdz. 2.2.7. W niniejszej pracy w algorytmie VAD przyjęto liczbę parametrów $p = 8$ w celu przyspieszenia obliczeń oraz zmniejszenia potrzebnej pamięci, choć w rekomendacji [110] zastosowano $p = 10$.

Energia we wszystkich pasmach E_f liczona jest zupełnie inaczej niż w (2.4) czy też (4.5), wykorzystując współczynnik autokorelacji $R(0)$ zgodnie z

$$E_f = 10\log_{10}[\frac{1}{10}R(0)]. \tag{4.11}$$

Kolejny parametr to energia niskich pasm E_l, definiowana jako

$$E_l = 10\log 10[\frac{1}{N}\mathbf{h}^T\mathbf{R}\mathbf{h}], \tag{4.12}$$

gdzie h jest odpowiedzią impulsową 12 rzędu filtru o skończonej odpowiedzi impulsowej FIR o częstotliwości odcięcia 1kHz, a R macierzą autokorelacji Toeplitza.

Ostatnim z parametrów jest współczynnik liczby przejść przez zero ZCR dla każdej z ramek. Jest on definiowany podobnie jak w (2.5) i (2.6) z tą różnicą, że dodatkowo jest normalizowany zgodnie z

$$ZCR = \frac{1}{2N}\sum_{i=0}^{N-1}[\,|\,\mathrm{sgn}[x(i)] - \mathrm{sgn}[x(i-1)]\,|], \tag{4.13}$$

gdzie N oznacza liczbę próbek w ramce.

Po wyznaczeniu parametrów dla każdej z ramek, należy porównać je z odpowiadającymi im średnimi parametrami szumu. Obliczane są one z pierwszych 100ms sygnału, który uważany jest za sygnał nie zawierający mowy, lecz tylko szum otoczenia.

Wyznaczone w ten sposób parametry dla każdej z ramek oznaczone zostały znakiem Δ lub słownie „delta” (na rysunkach).

I tak odpowiednio parametry ΔLSF definiowane są jako:

$$\Delta LSF = \sum_{i=1}^{p} (LSF_i - \overline{LSF}_i)^2 , \tag{4.14}$$

gdzie: $\overline{LSF}_i$ oznacza średnią wartość i-tego parametru LSF dla szumu.

Parametry Δ energii we wszystkich pasmach obliczano zgodnie z

$$\Delta E_f = \left| \overline{E}_f - E_f \right| , \tag{4.15}$$

gdzie $\overline{E}_f$ oznacza średnią energię szumu we wszystkich pasmach.

Parametry Δ energii niskich pasm, podobnie jak w (4.15), definiowano jako

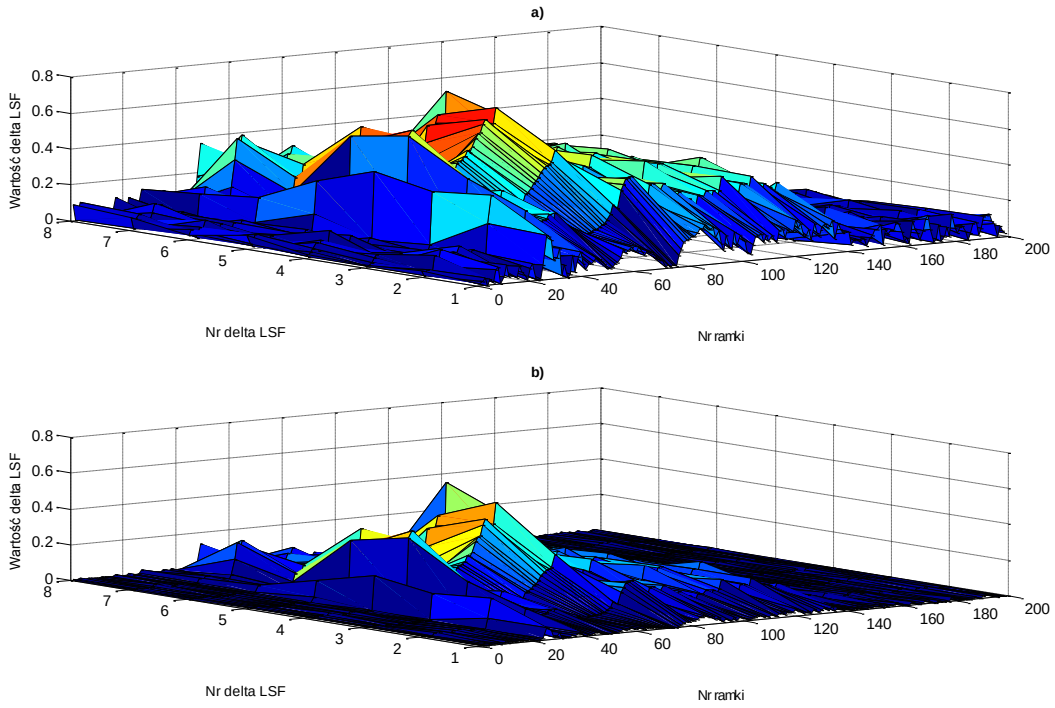
$$\Delta E_l = | \overline{E}_l - E_l | , \tag{4.16}$$

gdzie $\overline{E}_l$ jest średnią energią szumu niskich pasm.

Parametry ΔZCR zdefiniowano jako

$$\Delta ZCR = | \overline{ZCR} - ZCR | , \tag{4.17}$$

gdzie $\overline{ZCR}$ oznacza średnią liczbę przejść przez zero dla szumu.



Rys. 4.12 Wykresy parametrów ΔLSF przed (a) i po (b) operacji podniesienia do kwadratu

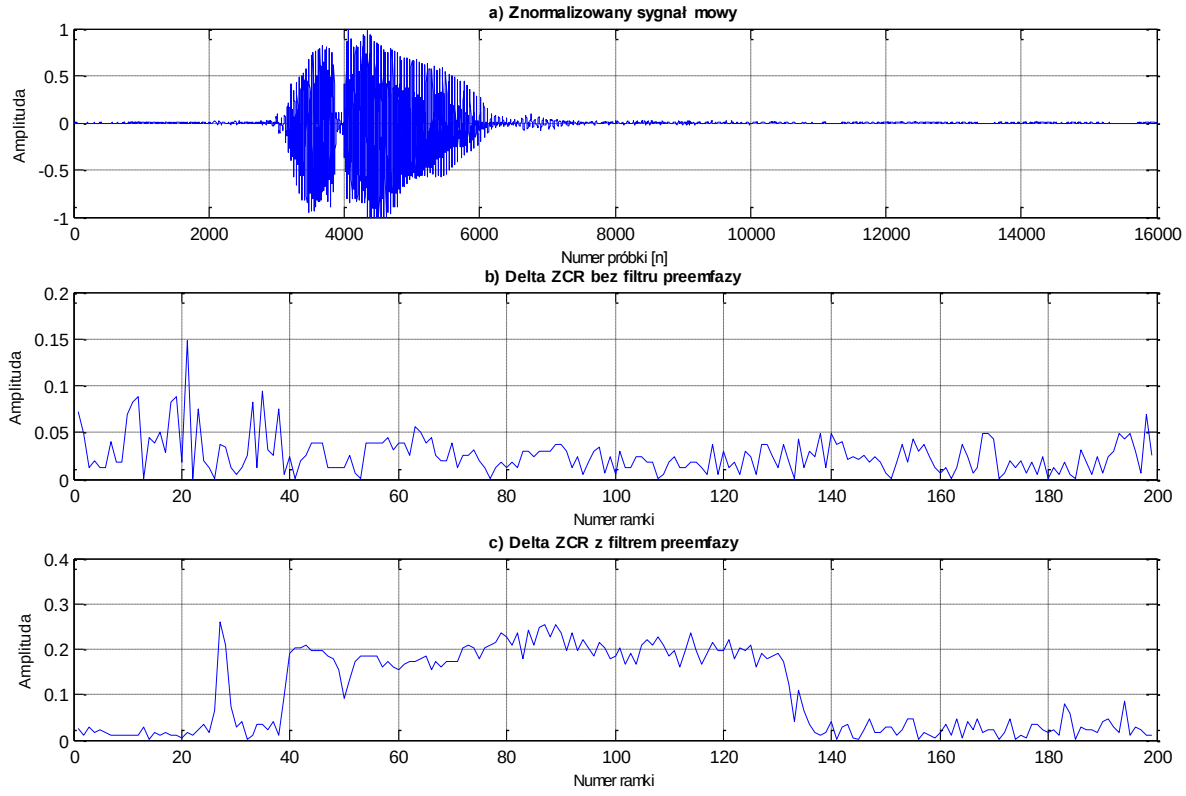
Na Rys. 4.12 przedstawiono różnicę parametrów ΔLSF dla poszczególnych ramek słowa *„faraon”* przed i po operacji podniesieniu do kwadratu. Jak można zauważyć na Rys. 4.12b), operacja podniesienia do kwadratu zgodnie z (4.14) jest jak najbardziej korzystna w szczególności w obecności szumu.

W rekomendacji G.729B [110] wybór początku i końca podejmowany jest na podstawie wielostopniowego kryterium decyzyjnego. W dalszej części pracy autor zaproponuje własne rozwiązanie dotyczące tego etapu działania algorytmu.

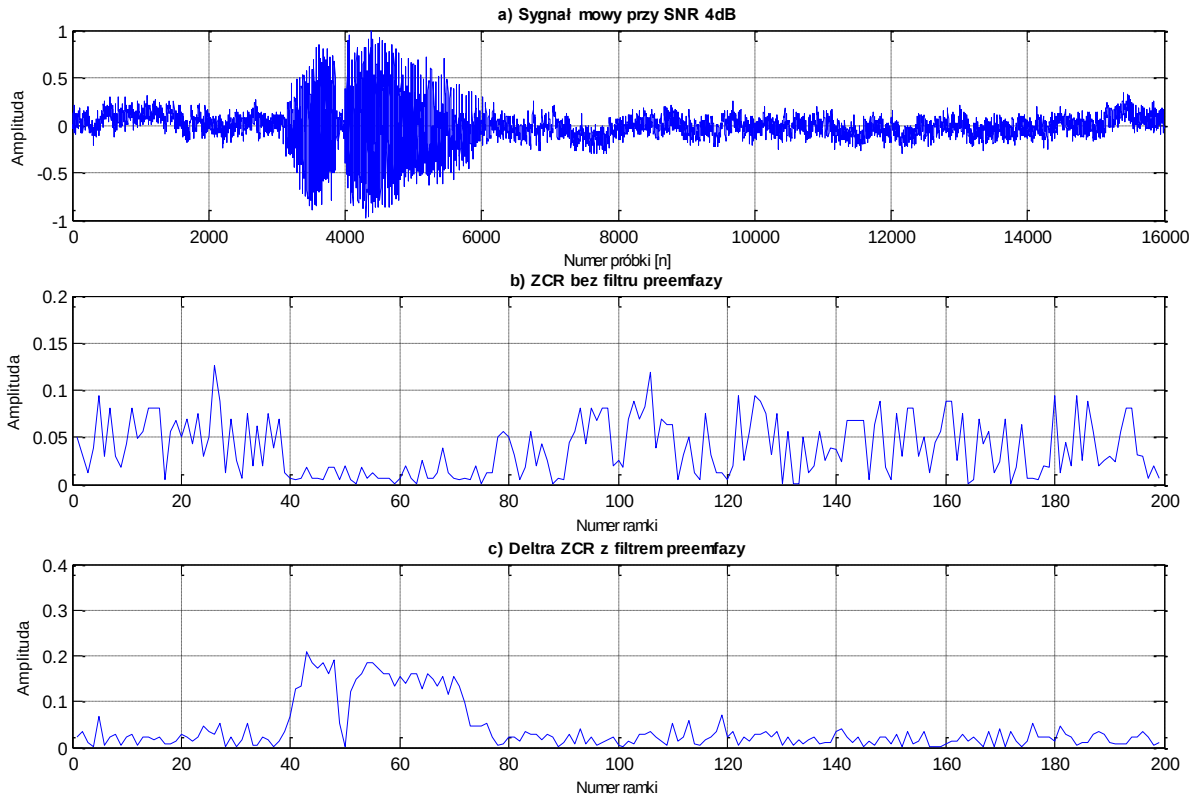
4.5.4 Wpływ filtru preemfazy na parametry ZCR

Przed przystąpieniem do dalszych badań autor postanowił zbadać także wpływ filtru preemfazy na parametry ΔZCR.

W tym celu wykonano testy w sytuacji, gdy: brak jest zewnętrznego szumu (Rys. 4.13) oraz przy SNR = 4dB (Rys. 4.14).



Rys. 4.13 Wpływ filtru preemfazy na parametry ZCR sygnału niezaszumionego dla słowa *„faraon”*: a) znormalizowany sygnał mowy, b) parametry ΔZCR bez filtru preemfazy, c) parametry ΔZCR z filtrem preemfazy



Rys. 4.14 Wpływ filtru preemfazy na parametry ZCR przy SNR = 4dB dla słowa *„faraon”*: a) zaszumiony znormalizowany sygnał mowy, b) parametry ΔZCR bez filtru preemfazy, c) parametry ΔZCR z filtrem preemfazy

Przy analizie uzyskanych wyników testów widać wyraźnie, że filtr preemfazy znacznie poprawia reprezentatywność sygnału mowy. Dużo wyraźniej widać obszary, gdzie mowa jest obecna. Jeśli nie zastosuje się preemfazy, to parametr ΔZCR staje się bezużyteczny nawet w przypadku braku zewnętrznego szumu (Rys. 4.13b).

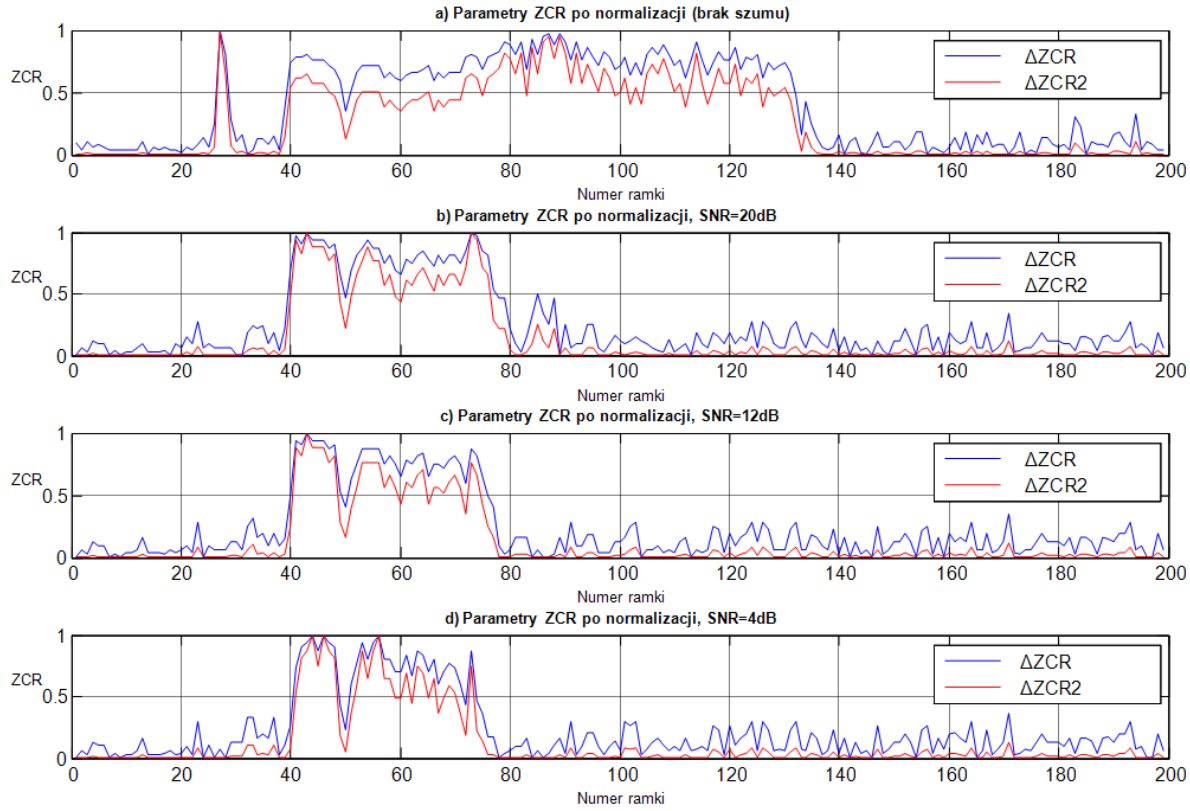
Podsumowując, w przypadku parametrów ΔZCR konieczne jest używanie filtru preemfazy, dlatego autor postanowił o jego zastosowaniu jeszcze przed uruchomieniem algorytmu VAD.

4.5.5 Wpływ operacji podniesienia do kwadratu parametrów ΔZCR na reprezentację sygnału mowy

Ze względu na fakt, że zwiększenie szumu pogorszało użyteczność parametru ΔZCR obliczanego zgodnie z (4.17), autor postanowił sprawdzić, jaki wpływ będzie miała operacja podniesienia do kwadratu zgodnie z zależnością

$$\Delta ZCR2 = (\overline{ZCR} - ZCR)^2 . \tag{4.18}$$

Testy porównawcze wykonano w przypadku: braku zewnętrznego szumu oraz przy SNR = 20dB, 12dB i 4dB dla słowa *„faraon”*. Otrzymane wyniki porównawcze zamieszczono na Rys. 4.15, oznaczając kolorem czerwonym parametry ΔZCR2 obliczone zgodnie z (4.18), oraz kolorem niebieskim parametry ΔZCR zgodnie z (4.17).



Rys. 4.15 Wpływ zastosowania operacji potęgowania na parametry ΔZCR dla słowa *„faraon”* w środowisku: a) braku zewnętrznego szumu, b) SNR = 20dB, c) SNR = 12dB oraz d) SNR = 4dB

Na Rys. 4.15 widać także, że w szumie wraz ze wzrostem poziomu zaszumienia gubione są te elementy, które reprezentują spółgłoski bezdźwięczne np. *„f”* czy też np. spółgłoski dźwięczne nosowe np. *„n”*. Praktycznie nie ma na to rady i mimo zastosowania operacji podniesienia do kwadratu nie da się temu zapobiec stosując parametry ΔZCR czy też ΔZCR2.

Wprowadzenie operacji podniesienia do kwadratu zgodnie z (4.18) po wcześniejszej normalizacji powoduje jednak, że wartości o większej amplitudzie tracą mniej na wartości niż te o większej amplitudzie. Dzięki temu wyraźnie zwiększa się różnica między ramkami reprezentującymi te części sygnału, w których obecna jest mowa od tych, w których jej w ogóle nie ma.

W związku z tym w dalszej części badań używana będzie jedynie forma ΔZCR2 zgodnie z (4.18).

4.5.6 Możliwość zastosowania parametrów VAD na etapie parametryzacji

Kolejnym zagadnieniem, które autor rozważał, było zastosowanie parametrów wykorzystywanych w algorytmie VAD na kolejnym etapie systemu rozpoznawania mowy, którym jest parametryzacja. Dzięki temu ilość niezbędnych obliczeń uległaby zmniejszeniu, powodując przyspieszenie działania całego systemu. Należy w tym przypadku odpowiedzieć na kilka pytań.

W dotychczasowych testach zakładano zgodnie z [53], że ramka sygnału ma 80 próbek, przy częstotliwości próbkowania 8kHz, co odpowiada długości 10ms. Ponadto ramki w tym przypadku nie zachodzą na siebie. Na etapie parametryzacji standardem są ramki o długości 25-30ms, choć

oczywiście zdarzają się wyjątki. Mimo że ramki często zachodzą na siebie zgodnie z (2.7), wydłużenie ramek spowoduje zmniejszenie ich liczby w porównaniu ze standardowym algorytmem VAD.

Ponadto zachodzenie na siebie nie wpływa w wyraźny sposób np. na zmianę funkcji energii. Daje się zaobserwować niewielkie wygładzenie funkcji przy dużej liczbie zachodzących na siebie próbek, lecz wydaje się, że nie ma to większego wpływu na skuteczność działania algorytmu.

Znacznie dłuższy czas obliczeń przy małym skoku oraz niewielki wpływ tego na wyniki pomiaru energii sygnału skłania ku zarzuceniu „*overlappingu*" w algorytmie wyboru początku i końca słowa.

Kolejne pytanie, które należy postawić, dotyczy samych parametrów. W przypadku rozważanego algorytmu VAD proponowane są parametry LSF, energii i gęstości przejść przez zero w różnych formach. Na etapie parametryzacji praktycznie tylko parametry LSF mogłyby być brane pod uwagę. Parametrów ZCR oraz energii praktycznie nie wykorzystuje się w tej części systemu. Co więcej, w algorytmie VAD nie korzysta się w sposób bezpośredni z parametrów LSF, tylko na ich podstawie oblicza się jedną wartość dla każdej z ramek, zgodnie z zależnością (4.14).

Ponadto parametry LSF mogą nienajlepiej reprezentować sygnał na etapie klasyfikacji z użyciem np. sieci typu C-SVC. Potwierdzają to późniejsze testy, których wyniki przedstawiono na Rys. 4.21 w podrozdz. 4.6.5. Podobnie parametry BFCC, które wstępnie wytypowano jako te, przy których poziom rozpoznawania jest największy (przy testach na bazie danych z oznaczonym wyborem początku i końca a priori), okazały się nie najlepiej reprezentatywne w algorytmie VAD.

W związku z powyższymi rozważaniami, autor zrezygnował z adaptacji parametrów z algorytmu VAD do dalszej części pracy, choć nie wyklucza, że w dalszych rozwinięciach systemu mających na celu jego optymalizację, można by było powrócić do tej kwestii.

4.5.7 Algorytm VAD oparty na sieciach SVM

Mając przygotowane parametry sygnału, przystąpiono do wyboru samego algorytmu VAD, który w skuteczny sposób wykorzysta parametry kilku typów do podjęcia decyzji o wyborze początku i końca. Jak stwierdzono wcześniej, proste kryteria decyzyjne oparte na jednym parametrze i progach, po przekroczeniu których następuje przyporządkowanie do danej klasy, nie zawsze są skuteczne. Autor skierował więc swe poszukiwania w innym kierunku. W ostatnich latach wzrosło zainteresowanie zastosowaniem sztucznych sieci neuronowych typu SVM właśnie do wyboru początku i końca sygnału.

Pierwotnie autor planował zastosowanie sieci typu C-SVC (ang. *C-Support Vector Classification*). Po głębszej analizie zrezygnowano z wykorzystania tego typu sieci przy opracowywaniu algorytmu VAD dla potrzeb niniejszej pracy. W sieciach tego typu, na etapie uczenia, jako dane wejściowe do uczenia modelu sieci podawane są wektory z danymi należącymi przynajmniej do dwóch klas wraz z informacją o przynależności do danej klasy. Mimo że algorytmy wykorzystujące sieci C-SVC do VAD istnieją [111], [112], to zdaniem autora trudno stwierdzić ich praktyczność w rzeczywistej pracy. Na etapie uczenia należałoby uczyć rozpoznawania szumu pochodzącego z różnych środowisk jak i różnych wypowiedzi jako sygnałów mowy. Nie bez znaczenia pozostaje także wzajemny stosunek sygnału do szumu, który niewątpliwie powinien być także wzięty pod uwagę na etapie uczenia.

Autor rozważał także wykorzystanie sieci C-SVC w inny sposób. Mianowicie w przypadku wyboru początku i końca, w odróżnieniu od zastosowania na etapie końcowej klasyfikacji w systemie rozpoznawania mowy, można założyć, że w pierwszych i np. ostatnich 100ms nagrywanego sygnału jest obecny tylko szum. W dalszym ciągu jednak brak jest informacji o drugiej klasie, czyli gdzie jest obecny już tylko sygnał mowy. Można byłoby zastosować wcześniej np. zmodyfikowany algorytm Rabinera-Sambura wyboru początku i końca, taki który zastosował autor niniejszej pracy w [108] lub też jego pewne modyfikacje, ale dalej wciąż brak gwarancji uzyskania poprawnej informacji o klasyfikacji całości sygnału mowy, w szczególności w środowisku zaszumionym. Gdyby taką informację uzyskano już na tym wstępnym etapie, nie byłoby już potrzeby stosowania dodatkowego algorytmu.

Zastosowanie sieci One-class SVM w algorytmie VAD

Logiczne staje się więc poszukiwanie takiego typu sieci, którą można by było uczyć rozpoznawania tylko jednej klasy, skoro nie można w sposób automatyczny uczyć na obu typach klas ze względu na problem automatycznej klasyfikacji typu „czy to jeszcze sygnał czy już szum" lub vice versa.

Często w algorytmach VAD przyjmuje się [16], że pierwszy fragment rejestrowanego sygnału (np. pierwsze 100ms) nie zawiera mowy, a jedynie szum otoczenia. Dzięki temu z dużym prawdopodobieństwem można uzyskać w automatyczny sposób poprawną informację o parametrach szumu. Znając parametry szumu w analizowanym okresie czasu, można je potraktować jako dane wejściowe do uczenia sieci przyporządkowując tym parametrom etykietę „szum". Po nauczeniu sieć jest w stanie dokonywać klasyfikacji, czy dana ramka sygnału zawiera szum, czy też nie.

Niewątpliwie największą zaletą tego rozwiązania jest jego prostota i szybkość działania. Zakładając, że sygnał mowy trwa jedną sekundę, a częstotliwość próbkowania wynosi 8kHz, to sygnał składa się z 8000 próbek dzielonych na ramki po np. 80. Daje to jedynie 100 ramek, z których każda reprezentowana jest jedynie przez 4 parametry: ΔLSF, ΔE_f, ΔE_l oraz ΔZCR opisane w podrozdz. 4.5.3. Daje to w sumie 100 punktów w przestrzeni czterowymiarowej.

Zadanie optymalnej klasyfikacji do dwóch klas (szum lub sygnał), danych w przestrzeni wielowymiarowej, z powodzeniem może realizować sieć neuronowa typu one-class SVM. Co więcej, ta sieć została zaprojektowana do rozwiązywania właśnie tego typu problemów [92].

Co ciekawe, autor nie znalazł informacji o wykorzystaniu sieci typu one-class SVM do wyboru początku i końca słowa. Jedyne źródło to chiński artykuł z 2006 r. [113] na serwerze o ograniczonym dostępie, do którego autorowi udało się zdobyć jedynie abstrakt.

Sieć tego typu, ze względu na małą liczbę danych wejściowych, bardzo szybko się uczy i szybko działa. Co więcej, proces uczenia odbywa się w oparciu o parametry szumu, który towarzyszy sygnałowi w danym czasie, a nie pełnej ich gamie, tak jak ma to miejsce np. w [112].

Opis algorytmu VAD bazującego na one-class SVM (VAD-1SVM)

Nowy algorytm wyboru początku i końca sygnału, wykorzystujący sieci neuronowe typu one class SVM, będzie w dalszej części pracy oznaczany skrótem **VAD-1SVM**.

Dzięki zastosowaniu filtru preemfazy (patrz podrozdz. 2.1.2) następuje podbicie składowych wyższych częstotliwości. Powoduje to lepszą reprezentację parametrów sygnału, takich jak np. parametry LPC, LSF, czy MFCC. Jak wykazano w podrozdz. 4.5.4, zastosowanie filtru preemfazy wpływa także korzystnie na sposób reprezentacji sygnału mowy na podstawie parametrów ZCR. Pierwszym więc krokiem poprzedzającym wybór początku i końca będzie przepuszczenie sygnału przez filtr preemfazy, zakładając a priori współczynnik $a = 0.95$.

Następnie, podobnie jak w przypadku innych algorytmów VAD [109], sygnał dzielony jest na ramki, które nie zachodzą na siebie, o długości 80 próbek, co przy 8kHz odpowiada 10ms sygnału.

Algorytm VAD-1SVM zakłada, że szum rejestrowany jest w pierwszych i ostatnich 100ms nagranego sygnału.

Z każdej z ramek obliczane są parametry LSF, E_f, E_l oraz ZCR całego nagranego sygnału a następnie parametry różnicowe $\Delta LSF, \Delta E_f, \Delta E_l$ oraz ΔZCR (patrz podrozdz. 4.5.3), które reprezentują ramki sygnału w przestrzeni czterowymiarowej.

Etap uczenia sieci

Uczenie sieci neuronowej typu one-class SVM odbywa się on-line dzięki małej ilości danych uczących. Jest to zaledwie 10 pierwszych i 10 ostatnich ramek sygnału, czyli tylko 80 liczb definiujących szum. Tworzą one 20 wektorów wejściowych, każdy o długości 4. Długość wektora wejściowego definiuje w tym przypadku wymiar przestrzeni.

Przed podaniem wektorów wejściowych na wejście sieci należy je poddać normalizacji. W sposób naturalny zakres normalizowanego sygnału powinien wynosić od 0 do 1, a nie od -1 do 1, ze względu na zastosowanie parametrów różnicowych. Oznacza to, że minimalne parametry Δ będą wartościami dodatnimi bliższymi zeru dla szumu oraz przyjmują większe wartości dla tych części sygnału, w których zawarta jest mowa.

Jako typ jądra wybrano jądro radialne zgodnie z wytycznymi zawartymi w [91]. Przyjęto a priori, że współczynnik γ (patrz Tab. 3.2) w jądrze wynosi $\frac{1}{k}$, gdzie k oznacza liczbę atrybutów w danych wejściowych.

Parametrem, który ma decydujący wpływ na jakość klasyfikacji (szum lub nie-szum) jest współczynnik $v \in (0,1]$. Dzięki niemu można regulować stopień czułości sieci na nowości. Im mniejszy jest ten współczynnik, tym sieć jest bardziej wrażliwa. Nie zawsze jest to jednak wskazane. Zbyt duża czułość może prowadzić do błędnej klasyfikacji przy większym szumie.

Wygładzanie wyników

Ze względu na fakt, że czasami niektóre pojedyncze ramki sygnału są klasyfikowane odmiennie niż ich najbliższe otoczenie, autor postanowił wprowadzić dodatkowo opcję „wygładzania" (ang. *smoothing*). Opcję wygładzania, choć realizowaną w inny sposób, autor znalazł w późniejszym czasie w [112], potwierdzając tym samym słuszność tej koncepcji.

Początkowo autor testował algorytm poszukujący wyników klasyfikacji pojedynczych ramek, które zostały inaczej zakwalifikowane niż ostatnia próbka przed oraz pierwsza po badanej ramce.

W przypadku wykrycia takich w sygnale, autor zmieniał ich znak (z -1 na 1 lub odwrotnie) na znak, pod którym zakwalifikowano najbliższe ramki przed nią i za nią.

W późniejszym okresie autor postanowił zaimplementować filtr medianowy do procesu wygładzania. Zgodnie z definicją **mediana** [107][13] (zwana też wartością środkową) to wartość cechy w szeregu uporządkowanym, powyżej i poniżej której znajduje się jednakowa liczba obserwacji.

Aby obliczyć medianę ze zbioru k próbek, należy je posortować w kolejności od najmniejszej do największej numerując od 1 do k. Następnie, jeśli k jest nieparzyste, medianą jest wartość próbki w środku o numerze $\frac{k+1}{2}$. Jeśli natomiast k jest parzyste, wynikiem jest średnia arytmetyczna dwóch środkowych próbek o numerach $\frac{k}{2}$ oraz $\frac{k+1}{2}+1$.

Filtr medianowy definiowany jest jako

$$Y(k) = MED[X(k-\frac{N-1}{2}:k+\frac{N-1}{2})] \text{ dla } N \text{ nieparzystych,} \tag{4.19}$$

oraz

$$Y(k) = MED[X(k-\frac{N}{2}:k+\frac{N}{2})] \text{ dla } N \text{ parzystych,} \tag{4.20}$$

gdzie:

k – wynik klasyfikacji po zastosowaniu one-class SVM dla k-tej ramki sygnału,

$\boldsymbol{X}$ – wektor z wynikami klasyfikacji po zastosowaniu one-class SVM,

$Y(k)$ – wynik filtracji z użyciem filtru medianowego,

MED – oznaczenie mediany,

: - oznaczenie zakresu.

W pracy przyjęto stopień filtru $N = 3$. W takim przypadku równanie (4.19) przyjmuje postać:

$$Y(k) = MED[X(k-1:k+1)] \tag{4.21}$$

W przypadku, gdy danego numeru ramki nie było, tzn. na początku i na końcu pojawiały się odwołania do pozycji spoza zakresu, uzupełniano je wtedy zerami.

Ze względu na fakt, że szum oznaczany jest liczbą -1, a sygnał cyfrą 1, zastosowano chwilową zamianę wartości -1 na 0, aby na wyjściu filtru medianowego pojawiały się wyłącznie 0 lub 1. Następnie w celu lepszej wizualizacji ramki sygnału o wartościach „0” zamieniono na wartości „-1”.

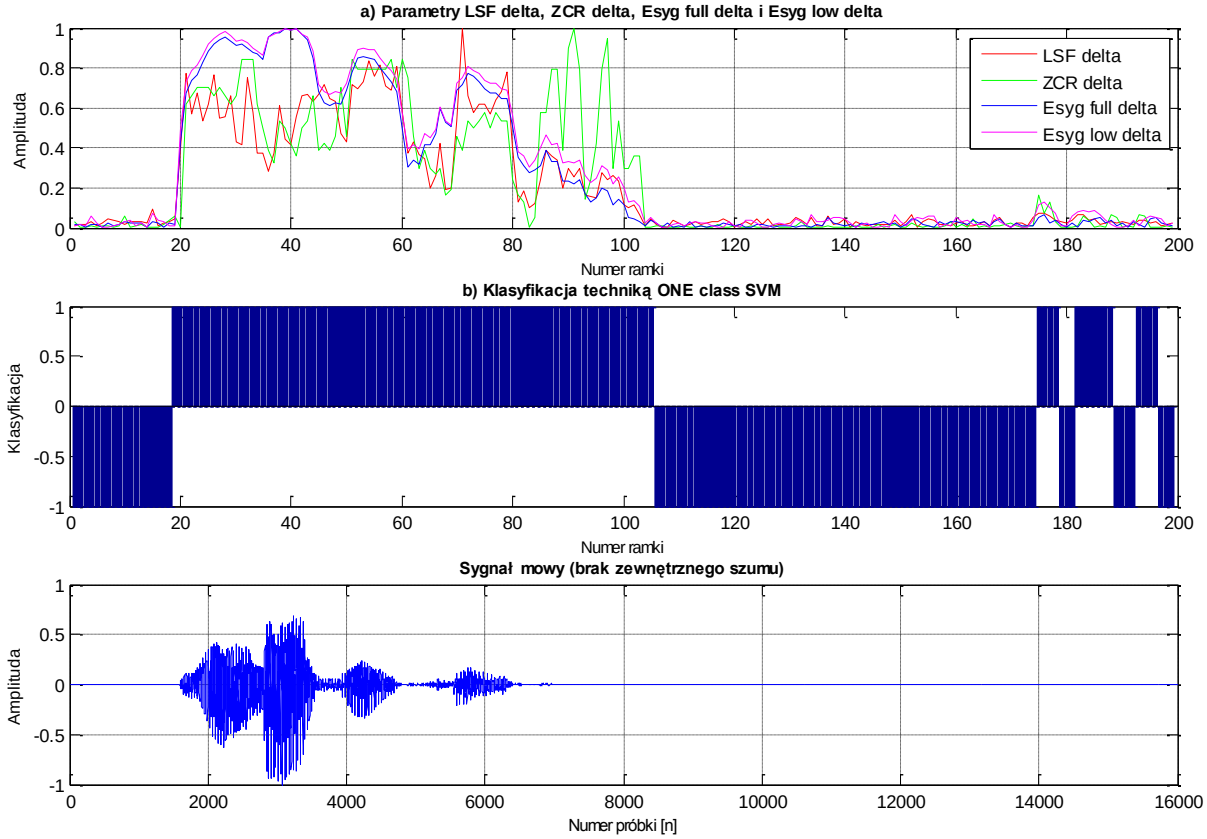
W czasie wstępnych testów okazało się, że wprowadzone kryterium może okazać się niewystarczające. Mogą pojawić się następujące problemy.

[13] hasło: mediana

Po pierwsze, w skrajnych przypadkach wszystkie ramki całej wypowiedzi mogą być zakwalifikowane do tej samej klasy, czyli cała wypowiedź może zostać zakwalifikowana jako szum lub nie-szum. Klasa nie-szum oznacza w tym przypadku sygnał mowy.

Kolejnym problemem może być występowanie, mimo wcześniej przeprowadzonego procesu wygładzania filtrem medianowym, zbyt dużej liczby grup ramek o różnych znakach. Oczywiście może się zdarzyć, że takie różnice wystąpią, ale ich zbyt duża ilość świadczy o błędnej klasyfikacji.

Następną sytuacją, która wymaga interwencji, jest przypadek, w którym mimo zastosowania filtru medianowego w ramkach na początku i końcu, które zakładano, że zawierają wyłącznie szum, były klasyfikowane jako „nie-szum". Przykład takiej sytuacji przedstawiono na Rys. 4.16, na którym koniec wypowiedzi, który powinien być rozpoznawany z definicji jako szum (ostatnie 100ms), jest rozpoznawany jako sygnał mowy.



Rys. 4.16 Przykład błędnego rozpoznania szumu jako sygnału wypowiedzi „*Ola Mucha*" w przypadku braku zewnętrznego szumu przy *v = 0,52*

Wszystkie powyższe wnioski skłoniły autora do wprowadzenia pewnych dodatkowych warunków, które muszą być spełnione, aby uznać, że otrzymane rezultaty klasyfikacji są poprawne.

Procedura uczenia i testowania w algorytmie VAD-1SVM

Wprowadzono procedurę uczenia i testowania sieci, która jest powtarzana iteracyjnie do momentu spełnienia następujących warunków:

- jeśli całą wypowiedź zakwalifikowano do jednej klasy, to powtórz procedurę uczenia i walidacji,

- lub, jeśli liczba grup ramek (grupa to przynajmniej 2 sąsiednie ramki zakwalifikowane do tej samej klasy) wynosi więcej niż 10, to powtórz procedurę uczenia i walidacji,

- lub, jeśli w ramkach reprezentujących pierwsze i ostatnie 100ms sygnału, które a priori ustalono, że zawierają tylko szum, były klasyfikowane jako nie-szum, to powtórz procedurę uczenia i walidacji.

W testach w podrozdz. 4.5.8 algorytm rozpoczyna procedurę uczenia z niskim współczynnikiem $v = 0.05$ i zwiększa jego wartość o 0.01 w przypadku niespełnienia któregoś z warunków. Ostatecznie jednak zdecydowano się na zwiększenie początkowej wartości współczynnika v do 0.15.

Wybór ramki początku i końca następuje po zakończeniu procedury uczenia i walidacji. Jako początek sygnału wybierana jest ta ramka sygnału, która jako pierwsza została zakwalifikowana jako nie-szum. Podobnie w przypadku końca sygnału. Ostatnia ramka, która została zakwalifikowana jako nie-szum uważana jest za tą, która określa koniec sygnału.

Mając dane numery ramek początku R_p i końca R_k, obliczano numery próbek początku i końca sygnału zgodnie z

$$p = (R_p - 1)(N_R - M_R), \tag{4.22}$$

oraz

$$k = R_k (N_R - M_R), \tag{4.23}$$

gdzie:

p – nr próbki początku,

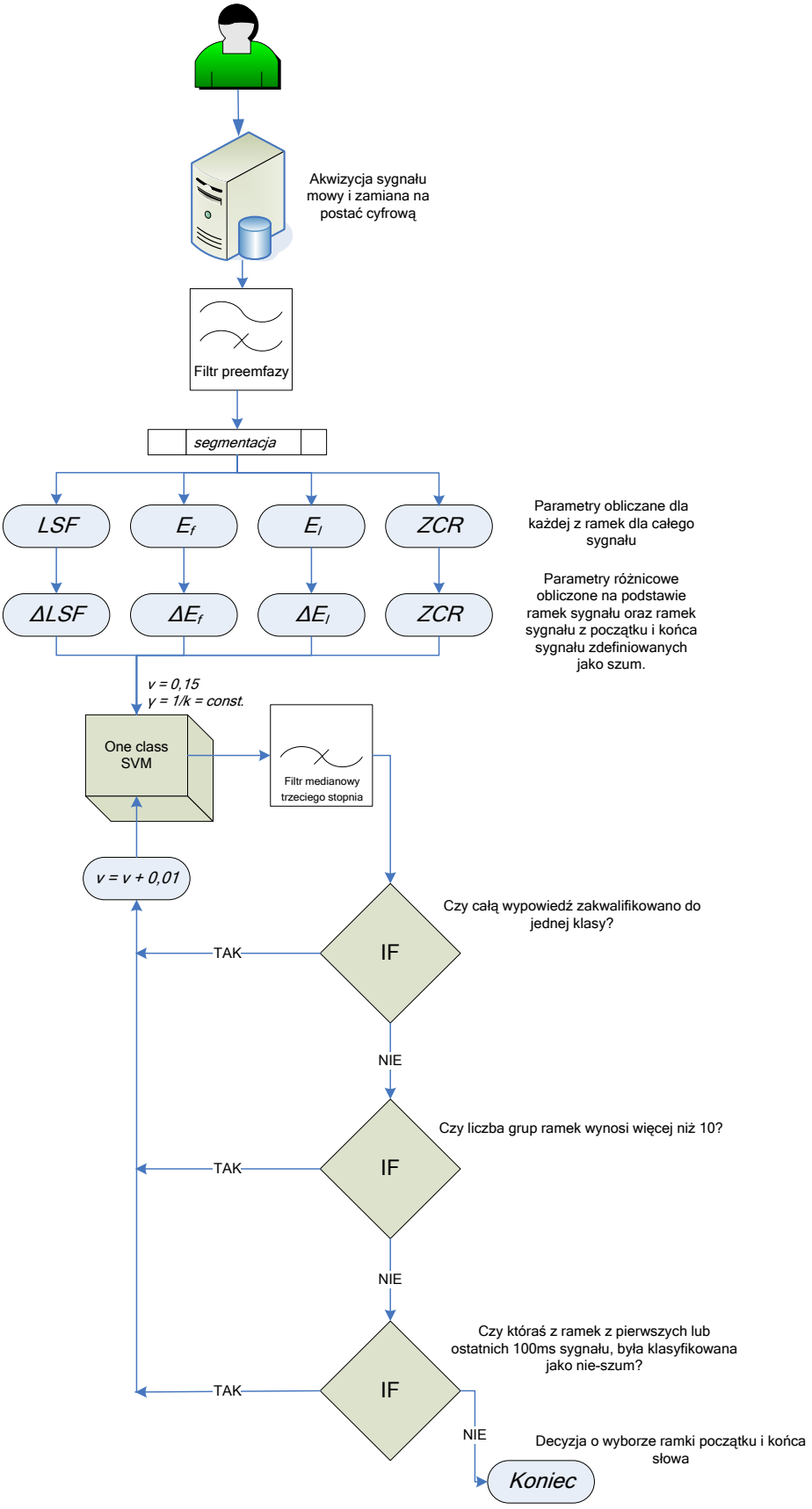
k – nr próbki końca,

N_R – długość ramki,

M_R – skok ramki.

W przypadku, gdy ramki nie nachodzą na siebie, skok wynosi 0 a wzory (4.22) i (4.23) ulegają dodatkowemu uproszczeniu.

Na Rys. 4.17 przedstawiono ogólny schemat działania algorytmu VAD-1SVM.



Rys. 4.17 Ogólny schemat działania algorytmu VAD-1SVM

4.5.8 Porównanie zmodyfikowanego algorytmu Rabinera-Sambura z algorytmem VAD-1SVM

Warunki eksperymentów

Testy porównawcze algorytmów wykonano z użyciem 10 plików dźwiękowych pochodzących z bazy CORPORA. Treść nagrań stanowiły cyfry: od *„zero”* do *„dziewięć”* (zapisane w plikach wav od SG1M1_0 do SG1M1_9, nagrane przez mężczyznę w wieku pomiędzy 30 a 50 lat). Ze względu na fakt, że nie zawsze w bazie CORPORA początek nagrania był tożsamy z początkiem wypowiedzi, niektóre z nagrań skrócono specjalnie dla tego testu, dopasowując tym samym początek sygnału do początku wypowiedzi. W żadnym z plików nie było potrzeby modyfikacji końca nagrania. Przed wykonaniem właściwych testów każdy z sygnałów został przepróbkowany z 16 do 8kHz ze względu na fakt wyboru takiej właśnie częstotliwości próbkowania dla całego systemu.

Pliki dźwiękowe z bazy CORPORA powinny zawierać wyłącznie sygnał mowy po wcześniejszym ręcznym wyborze początku i końca wykonanym przez twórców bazy. W celu umożliwienia sprawdzenia działania algorytmów, dodano do każdego z testowanych plików próbki z wartością 0 - po 150ms przed sygnałem jak i za jego końcem. W przypadku $Fs = 8kHz$ odpowiada to wydłużeniu testowanego sygnału o 1200 próbek z przodu i taką samą liczbę próbek na jego końcu.

Oba algorytmy porównywano przy SNR równym 100dB, 45dB, 40dB, 36dB, 28dB, 20dB, 12dB, 10dB i 8dB. Dodatkowo dla algorytmu VAD-1SVM wykonano testy także w środowisku bardzo zaszumionym przy SNR wynoszącym 4dB, 0dB a nawet -4dB.

Należy zaznaczyć, że szum dla każdego z sygnałów generowany jest osobno w sposób losowy. W niektórych przypadkach może to powodować, że wyniki skuteczności badanych algorytmów mogą się od siebie różnić. Postanowiono więc, aby liczba powtórzeń N każdego testu wynosiła cztery. Następnie na podstawie błędów policzono ich wartość średnią oraz odchylenie standardowe.

Przy testach skuteczności wyznaczania początku słowa korzystano z następujących definicji:

$$\Delta_p^i = \frac{\left| p_i - \overline{p}_i \right|}{n_i} 100\% , \tag{4.24}$$

$$\overline{\Delta}_p^j = \frac{1}{m} \sum_{i=1}^{m} \Delta_p^i \quad , \tag{4.25}$$

$$\overline{\Delta}_p = \frac{1}{N} \sum_{j=1}^{N} \overline{\Delta}_p^j , \tag{4.26}$$

$$\delta_p = \sqrt{\frac{1}{N-1} \sum_{j=1}^{N} (\Delta_p^j - \overline{\Delta}_p)^2} , \tag{4.27}$$

gdzie:

Δ_p^i – błąd wyboru początku i-tego słowa,

p_i – nr próbki początku i-tego słowa obliczony z użyciem algorytmu VAD,

$\overline{p}_i$ – pierwszy nr próbki i-tego słowa z bazy CORPORA,

n_i – liczba próbek i-tego oryginalnego sygnału,

$\overline{\Delta}_p^j$– średni błąd wyboru początku m słów dla j-tego powtórzenia,

$\overline{\Delta}_p$ – średni błąd wyboru początku m słów dla N powtórzeń,

m – liczba sygnałów poddawana testom,

δ_p – odchylenie standardowe błędów wyboru początku testowanych słów dla N powtórzeń,

N – liczba powtórzeń danego testu ($N = 4$).

Analogiczne definicje błędów i odchylania standardowego zdefiniowano dla końca wypowiedzi:

$$\Delta_k^i = \frac{\left|k_i - \overline{k}_i\right|}{n_i} 100\%, \tag{4.28}$$

$$\overline{\Delta}_k^j = \frac{1}{m}\sum_{i=1}^{m} \Delta_k^i, \tag{4.29}$$

$$\overline{\Delta}_k = \frac{1}{N}\sum_{j=1}^{N} \overline{\Delta}_k^j, \tag{4.30}$$

$$\delta_k = \sqrt{\frac{1}{N-1}\sum_{j=1}^{N} (\Delta_k^j - \overline{\Delta}_k)^2}, \tag{4.31}$$

gdzie dodatkowo:

Δ_k^i – błąd wyboru końca i-tego słowa,

k_i – nr próbki końca i-tego słowa obliczony z użyciem algorytmu VAD,

$\overline{k}_i$ – ostatni nr próbki i-tego słowa z bazy CORPORA ($\overline{k}_i = n_i$),

$\overline{\Delta}_k^j$– średni błąd wyboru końca m słów dla j-tego powtórzenia,

$\overline{\Delta}_k$ – średni błąd wyboru końca m słów dla N powtórzeń,

δ_k – odchylenie standardowe błędów wyboru końca testowanych słów dla N powtórzeń.

Na podstawie błędów początku i końca zdefiniowano błąd całkowity i odchylenie standardowe dla całej wypowiedzi zgodnie z:

$$\Delta_c^i = \Delta_p^i + \Delta_p^i = \frac{\left|p - \bar{p}_i\right| + \left|k_i - \bar{k}_i\right|}{n_i} 100\%\,, \tag{4.32}$$

$$\overline{\Delta}_c^j = \overline{\Delta}_p^j + \overline{\Delta}_k^j = \frac{1}{m}\sum_{i=1}^{m}(\Delta_p^i + \Delta_k^i)\,, \tag{4.33}$$

$$\overline{\Delta}_c = \frac{1}{N}\sum_{j=1}^{N}\overline{\Delta}_c^j \tag{4.34}$$

oraz

$$\delta_c = \sqrt{\frac{1}{N-1}\sum_{j=1}^{N}(\Delta_c^j - \overline{\Delta}_c)^2}\,, \tag{4.35}$$

gdzie:

$\overline{\Delta}_c^j$– średni całkowity błąd wyboru początku i końca m słów dla j-tego powtórzenia,

$\overline{\Delta}_c$ – średni błąd wyboru początku i końca m słów dla N powtórzeń,

δ_c – odchylenie standardowe błędów wyboru końca testowanych słów dla N powtórzeń.

Wyniki testów porównawczych

Wyniki średnich błędów dla każdego z powtórzeń zestawiono w Tab. 4.1.

Tab. 4.1 Wyniki testów porównawczych algorytmu MRSED i VAD-1SVM

SNR [dB]	Algorytm MRSED [%] $\overline{\Delta}_p^j$	$\overline{\Delta}_k^j$	$\overline{\Delta}_c^j$	Algorytm VAD-1SVM [%] $\overline{\Delta}_p^j$	$\overline{\Delta}_k^j$	$\overline{\Delta}_c^j$	SNR [dB]	Algorytm MRSED [%] $\overline{\Delta}_p^j$	$\overline{\Delta}_k^j$	$\overline{\Delta}_c^j$	Algorytm VAD-1SVM [%] $\overline{\Delta}_p^j$	$\overline{\Delta}_k^j$	$\overline{\Delta}_c^j$
100	0,00	0,60	0,60	4,13	4,33	8,46	12	24,48	49,08	73,56	11,04	9,71	20,75
100	0,00	0,60	0,60	1,69	2,63	4,32	12	24,52	44,10	68,62	13,63	13,82	27,45
100	0,00	0,60	0,60	3,01	3,41	6,42	12	24,33	46,78	71,11	8,18	10,64	18,82
100	0,00	0,60	0,60	4,35	5,99	10,34	12	24,58	46,89	71,47	8,93	12,90	21,83
45	1,97	4,20	6,17	3,60	5,55	9,15	10	25,50	54,94	80,44	14,53	13,82	28,35
45	1,97	4,03	6,00	2,57	3,64	6,21	10	25,26	59,04	84,30	12,67	14,40	27,07
45	1,97	4,38	6,35	3,00	3,67	6,67	10	25,12	59,63	84,75	11,51	15,56	27,07
45	2,26	4,18	6,44	2,73	4,50	7,23	10	25,14	55,27	80,41	14,65	11,07	25,72
40	2,77	7,09	9,86	2,85	3,98	6,83	8	24,76	75,28	100,04	15,17	14,75	29,92
40	2,63	7,20	9,83	2,85	4,47	7,32	8	30,28	73,53	103,81	12,50	13,78	26,28
40	2,51	7,20	9,71	6,22	6,41	12,62	8	23,90	76,13	100,03	16,63	10,50	27,13
40	2,77	7,06	9,83	3,55	7,46	11,02	8	25,41	75,29	100,70	15,62	17,70	33,32
36	3,05	8,86	11,91	5,81	6,53	12,34	4	-	-	-	15,35	21,24	36,59
36	3,73	8,71	12,45	3,55	6,05	9,61	4	-	-	-	14,27	20,77	35,04
36	3,29	8,71	12,00	3,00	4,79	7,78	4	-	-	-	14,56	21,97	36,53
36	3,05	8,73	11,78	3,99	7,12	11,11	4	-	-	-	19,55	19,49	39,04
28	10,10	12,86	22,97	4,48	6,33	10,81	0	-	-	-	17,59	24,94	42,53
28	10,83	13,28	24,11	4,61	5,34	9,95	0	-	-	-	21,03	27,08	48,11
28	11,12	14,57	25,69	4,88	8,11	12,99	0	-	-	-	19,85	28,15	48,00
28	10,25	12,48	22,73	4,35	6,47	10,82	0	-	-	-	17,86	18,62	36,48
20	19,06	25,32	44,38	7,46	8,85	16,31	-4	-	-	-	11,59	27,16	38,75
20	18,94	25,47	44,41	4,70	7,55	12,25	-4	-	-	-	23,39	32,65	56,04
20	19,03	25,82	44,85	7,10	7,85	14,95	-4	-	-	-	20,17	27,00	47,17
20	18,03	26,01	44,04	8,92	10,19	19,11	-4	-	-	-	21,05	42,28	63,33

Wszystkie wyniki w Tab. 4.1, Tab. 4.2 i Tab. 4.3 przedstawiono w procentach.

Tab. 4.2 Błąd średni skuteczności działania algorytmu MRSED przy różnym SNR

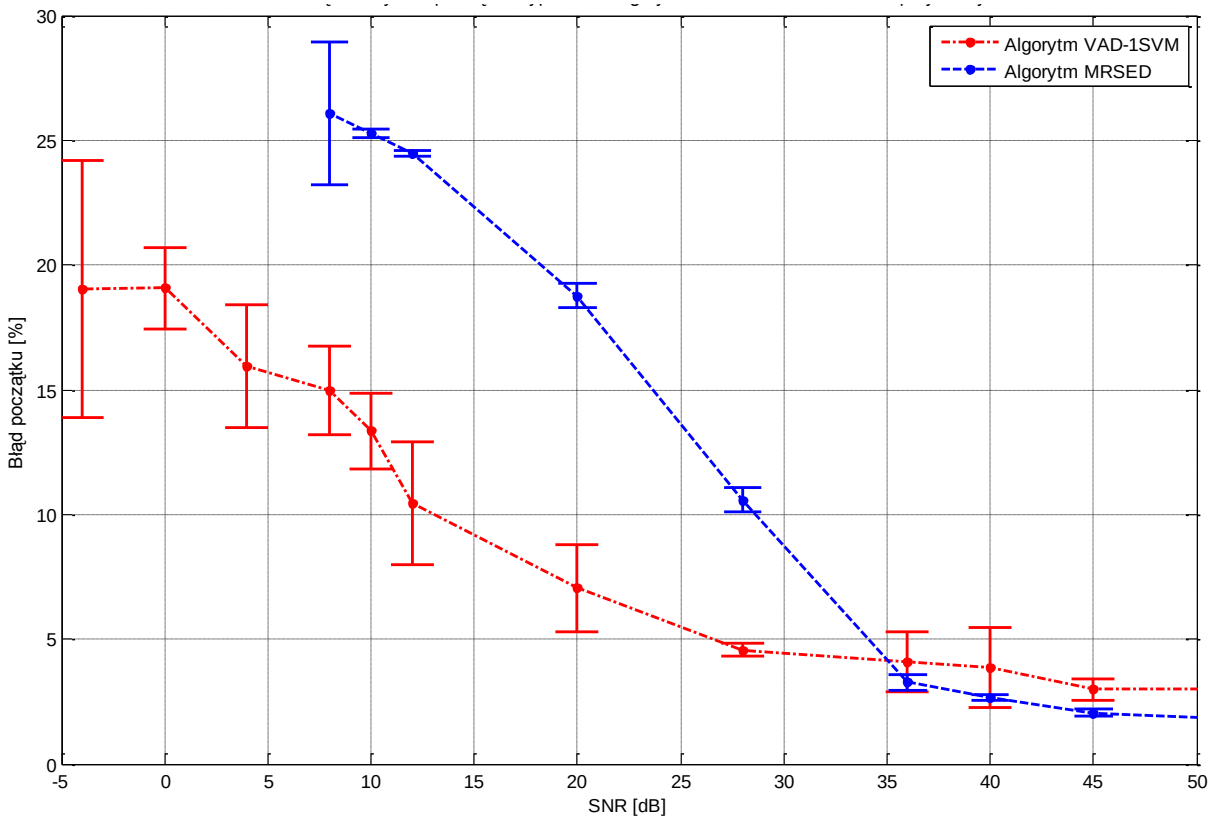
SNR [dB]	$\overline{\Delta}_p$ [%]	δ_p [%]	$\overline{\Delta}_k$ [%]	δ_k [%]	$\overline{\Delta}_c$ [%]	δ_c [%]
100	0,00	0,00	0,60	0,00	0,60	0,00
45	2,04	0,15	4,20	0,14	6,24	0,20
40	2,67	0,12	7,14	0,07	9,81	0,07
36	3,28	0,32	8,76	0,07	12,04	0,29
28	10,58	0,48	13,30	0,91	23,88	1,35
20	18,77	0,49	25,66	0,32	44,42	0,33
12	24,48	0,11	46,71	2,04	71,19	2,03
10	25,26	0,17	57,22	2,46	82,48	2,37
8	26,09	2,86	75,06	1,09	101,14	1,80

Tab. 4.3 Błąd średni skuteczności działania algorytmu VAD-1SVM przy różnym SNR

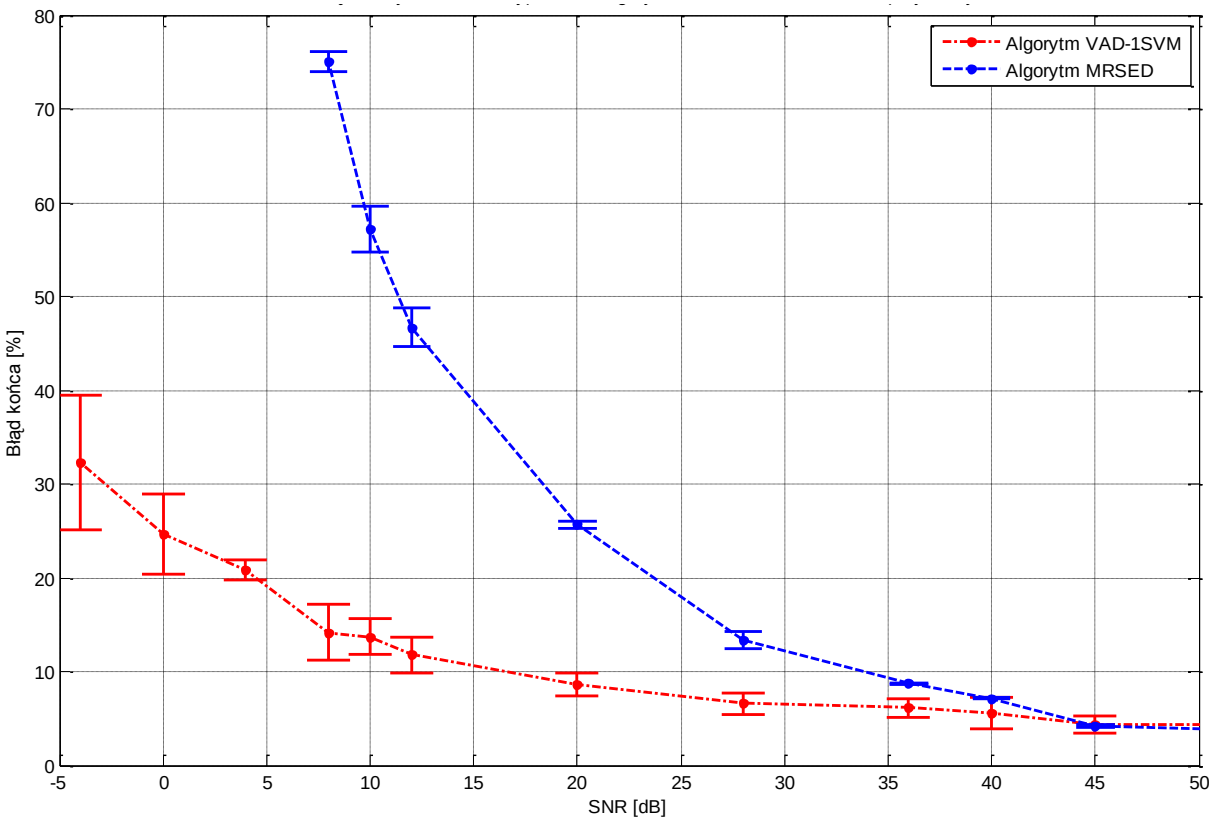
SNR [dB]	$\overline{\Delta}_p$ [%]	δ_p [%]	$\overline{\Delta}_k$ [%]	δ_k [%]	$\overline{\Delta}_c$ [%]	δ_c [%]
100	3,30	1,22	4,09	1,44	7,39	2,60
45	2,98	0,45	4,34	0,90	7,32	1,29
40	3,87	1,60	5,58	1,64	9,45	2,82
36	4,09	1,22	6,12	0,99	10,21	1,97
28	4,58	0,23	6,57	1,15	11,14	1,30
20	7,05	1,75	8,61	1,19	15,66	2,86
12	10,45	2,44	11,77	1,91	22,21	3,71
10	13,34	1,52	13,71	1,90	27,05	1,07
8	14,98	1,76	14,18	2,97	29,16	3,18
4	15,93	2,45	20,87	1,04	36,80	1,66
0	19,08	1,64	24,70	4,27	43,78	5,52
-4	19,05	5,16	32,27	7,17	51,32	10,67

Wizualizacja otrzymanych wyników

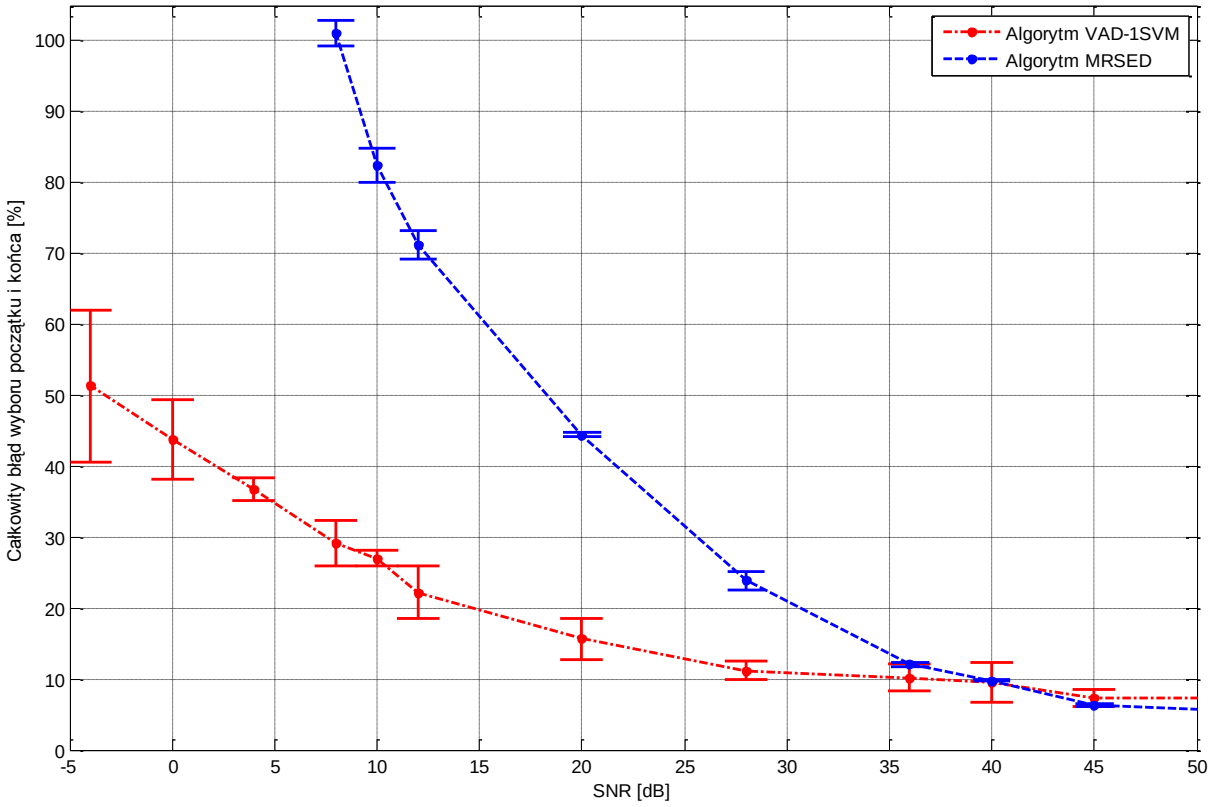
Na podstawie wyników zamieszczonych w Tab. 4.2 oraz Tab. 4.3 wykonano wykresy porównawcze obu algorytmów.



Rys. 4.18 Porównanie błędów wyboru początku wypowiedzi algorytmów MRSED i VAD-1SVM przy różnym SNR



Rys. 4.19 Porównanie błędów wyboru końca wypowiedzi algorytmów MRSED i VAD-1SVM przy różnym SNR



Rys. 4.20 Porównanie całkowitych błędów wyboru początku i końca wypowiedzi algorytmów MRSED i VAD-1SVM przy różnym SNR

4.5.9 Interpretacja wyników testów i podsumowanie badań nad algorytmami wyboru początku i końca słowa

Z analizy tabel i wykresów z podrozdz. 4.5.8 można wysunąć szereg wniosków.

Zauważono, że algorytm MRSED nie mógł funkcjonować przy SNR $\leq$ 8dB, gdy algorytm VAD-1SVM w takim środowisku mógł nadal działać. Nie oznacza to jednak, że algorytm MRSED w każdym przypadku nie mógł funkcjonować. Wystarczyło jednak, żeby algorytm raz nie zadziałał w czasie eksperymentu, aby uznać, że algorytm nie był w stanie wykonać testu.

Oba algorytmy zdecydowanie lepiej radziły sobie z wyborem początku sygnału niż jego końca. W przypadku algorytmu VAD-1SVM poziom różnicy błędów pomiędzy wyborem początku a końca był rzędu kilkudziesięciu procent, gdy w przypadku algorytmu MRSED różnica pomiędzy skutecznością wyboru początku i końca wynosiła nawet kilkaset procent (maksymalnie prawie 290% przy SNR = 8dB - patrz Tab. 4.2).

Wyniki skuteczności algorytmu VAD-1SVM charakteryzują się dużo większym odchyleniem standardowym niż w przypadku algorytmu MRSED.

Oba algorytmy mają podobną skuteczność przy SNR ≥ 36dB. Trudno jednak przy takim SNR mówić o środowisku zaszumionym. Jeśli szum jest większy, widać bardzo wyraźną i szybko rosnącą różnicę w skuteczności działania obu algorytmów. W przypadku SNR = 20dB całkowity błąd algorytmu VAD-1SVM wyniósł niespełna 15,7% przy odchyleniu standardowym poniżej 3%. Dla porównania, błąd algorytmu MRSED wyniósł ponad 44% przy niskim odchyleniu standardowym na poziomie 0,3%. Nawet zakładając na korzyść algorytmu MRSED odchylenie standardowe z obu pomiarów, otrzymana różnica w skuteczności obu algorytmów przy SNR=20dB wynosi ponad 138%.

Podsumowanie

Algorytm VAD-1SVM charakteryzuje się znacznie większą skutecznością rozpoznawania w stosunku do algorytmu MRSED w środowisku zaszumionym. Jednocześnie oba algorytmy mają podobną skuteczność w przypadku braku zewnętrznego szumu. Ze względu na fakt większego skomplikowania algorytmu VAD-1SVM logiczne jest zaproponowanie rozwiązania hybrydowego, które w zależności od stosunku SNR uruchomi bądź to algorytm MRSED przy SNR $\geq$ 36dB, lub też wybierze algorytm VAD-1SVM w przypadku większego zaszumienia.

W stosunku do parametrów z rekomendacji G729-B wprowadzono modyfikację polegającą na zmianie parametrów różnicowych ZCR. Dzięki zastosowaniu operacji podnoszenia do kwadratu zmniejszono wpływ szumu na sposób reprezentacji tej części sygnału, która zawiera mowę.

4.6 Zastosowanie sieci typu SVM do klasyfikacji i końcowe testy systemu

W podrozdziale zostaną przedstawione wyniki badań, które mają na celu pomóc w podjęciu decyzji o doborze ostatecznych parametrów projektowanego systemu izolowanych słów. Wszystkie eksperymenty (poza podrozdz. 4.6.15) wykonano także w środowisku zaszumionym przy różnym SNR.

Niewątpliwie testy powinny dać odpowiedź, które z metod parametryzacji należy zastosować w systemie (patrz podrozdz. 4.6.4 i 4.6.5). Zbadany zostanie również, w podrozdz. 4.6.6, wpływ długości wektorów parametrów, które są podawane na wejście sieci SVM zarówno na etapie uczenia, jak i późniejszej normalnej pracy systemu. Dodatkowo, w osobnych testach w podrozdz. 4.6.7, porównana zostanie metoda uzupełnienia zerami z metodą stałej liczby próbek.

Kolejne trzy grupy eksperymentów związane są ściśle z techniką wektorów podtrzymujących. Wiadomo, że w sieciach SVM ważny jest odpowiedni dobór stałej C i parametru jądra γ. Wpływ tych parametrów na wyniki rozpoznawania zostanie zbadany w teście w podrozdz. 4.6.8. W kolejnych badaniach w podrozdz. 4.6.9 badany będzie wpływ skalowania danych oraz metody *cross-validation* w połączeniu z *grid-search* na wyniki rozpoznawania. Celem ostatniego z eksperymentów z tej grupy (patrz podrozdz. 4.6.10) jest zbadanie, jaki wpływ na skuteczność rozpoznawania ma typ jądra, które zostanie użyte w sieci SVM.

Kolejne badanie pokazuje, jaki wpływ na skuteczność systemu ma liczba klas, spośród których podejmowana jest decyzja o klasyfikacji. Wiadomo, że im więcej klas, tym trudniej jest podjąć poprawną decyzję o klasyfikacji. W teście w podrozdz. 4.6.11 zbadano, jaki rzeczywisty ma to wpływ na projektowany system rozpoznawania izolowanych słów z użyciem sieci SVM.

Testy w podrozdz. 4.6.12 są nieco podobne do tych z podrozdz. 4.6.11. Różnica polega na tym, że tym razem zmieniana jest liczba powtórzeń każdej z klasy na etapie uczenia, a następnie badana jest skuteczność całego systemu, dzięki czemu można dobrać optymalną ich liczbę do uczenia sieci SVM.

Następne testy w podrozdz. 4.6.13 porównują oba algorytmy wyboru początku i końca zastosowane w projektowanym systemie rozpoznawania mowy i wskazują, przy jakim SNR ma zastosowanie każdy z algorytmów.

Dwie ostatnie grupy testów mają na celu zbadanie skuteczności systemu przy założeniu, że będzie z niego korzystać wielu mówców. W pierwszej grupie z podrozdz. 4.6.14 badane jest np., w jakim stopniu zawężenie grupy mówców tylko do głosów męskich wpływa na wyniki skuteczności systemu.

Ostatni z eksperymentów (patrz podrozdz. 4.6.15) miał na celu porównanie stworzonego systemu z systemem rozpoznawania mowy z użyciem ukrytych modeli Markowa. Ze względu na porównanie z istniejącymi wynikami uzyskanymi w [106] nie zastosowano algorytmu VAD.

4.6.1 Wyjaśnienie oznaczeń stosowanych w opisach testów

W Tab. 4.4 przedstawiono opis oznaczeń stosowanych przy opisie zrealizowanych badań.

Tab. 4.4 Opis oznaczeń stosowanych przy opisie testów

Baza Danych	Oznacza nazwę bazy danych sygnałów mowy (CORPORA lub WBI).
Etykiety	Określa temat rozpoznawania np. cyfry, imiona.
Liczba klas	Definiuje liczbę klas rozpoznawania, np. 10 różnych cyfr, każda tworząca osobną klasę.
Uczenie [liczba plików]	Określa całkowitą liczbę sygnałów mowy, które zastosowano na etapie uczenia.
Testy [liczba plików]	Określa całkowitą liczbę sygnałów mowy, które zastosowano na etapie testowania.
Rozdz. pomiaru	Określa rozdzielczość pomiaru, definiowaną jako odwrotność liczby sygnałów mowy, biorących udział na etapie testowania, np. dla 100 sygnałów uczących wyniesie 1/100 = 1%.
Skalowanie	Określa, do jakiego zakresu przeskalowano dane. Najczęściej jest to zakres od -1 do 1 oznaczany jako [-1,1]. W niektórych testach skalowanie przeprowadzono w zakresie [0,1].
Typ jądra	Najczęściej jest to jądro radialne. W jednym z testów testowane są ponadto jądra: wielomianowe, sigmoidalne i liniowe.
LDCV	Oznacza zastosowanie metody v-*fold cross-validation* połączonej z metodą *grid-search*. Określa, czy zastosowano na etapie uczenia metodę *cross-validation* w połączeniu z *grid-search,* a jeśli tak, to na ile podzbiorów dzielony był zbiór uczący. W większości przypadków, w których zastosowano *cross-validation*, wektory były dzielone na 5 równych podgrup (w kilku przypadkach także na 9 podgrup). Jeśli zastosowano *cross-validation*, to stosowano także w tym przypadku metodę *grid-search*.
CVGS [%]	Określa wynik skuteczności uczenia sieci SVM po zastosowaniu metody *cross-validation* i *grid-search*. Skuteczność w tym przypadku jest badana na wejściowym zbiorze uczącym jeszcze przed testowaniem sieci na innych danych testowych.
SV	Liczba wektorów podtrzymujących, określona po zakończeniu etapu uczenia sieci SVM.
WR(xxdB)[%]	Oznacza procentowy wynik rozpoznawania, przy określonym stosunku sygnału do szumu.
log2(γ)	Określa wartość parametru γ w postaci logarytmu o podstawie 2.
log2(C)	Określa wartość parametru C w sieci C-SVC w postaci logarytmu o podstawie 2.
$Fp\ [kHz]$	Określa częstotliwość w kilohercach, przy której były wykonywane testy.
Met. parametr.	Określa **metodę parametryzacji**. W większości testów używano BFCC. W niektórych testach korzystano również z: MFCC, ISP, LPCC, RC, LAR, LPC, LSF (patrz podrozdz. 2.2)
Algorytm VAD	Opisuje, jaki wybrano algorytm wyboru początku i końca słowa. Możliwe opcje to: *VAD-1SVM, MRSED* oraz *„Brak".* To ostatnie oznaczenie wskazuje, że w teście nie korzystano z algorytmu VAD. Jeśli wybrano algorytm VAD-1SVM, to domyślnie zakłada się, że wartość początkowa współczynnika v w tym algorytmie wynosi $0{,}15$. Jeśli było inaczej, opisano to w nawiasie, np. *VAD-1SVM* ($v = 0{,}05$).
k	Oznacza długość wektora parametrów. W niektórych testach, gdzie nie korzystano z metody *cross-validation* ani *grid-search*, parametr k wpływa na domyślną wartość stałej C.
SN	Jest to oznaczenie szumu naturalnego

4.6.2 Ustawienia systemu stosowane w testach

Zanim przystąpiono do testów poszczególnych elementów systemu rozpoznawania mowy, należało wstępnie wybrać metody i parametry systemu poddawanego testom. Pokrótce zostanie więc opisany system rozpoznawania mowy, który wykorzystywano w czasie realizacji badań.

Pierwszym etapem systemu rozpoznawania mowy jest blok przetwarzania wstępnego, w którym można wyróżnić etap akwizycji sygnału, preemfazy, wyboru początku i końca słowa oraz ramkowania i okienkowania sygnału mowy. Na etapie akwizycji sygnału standardową częstotliwość próbkowania autor ustalił na 8kHz. W niektórych testach z użyciem bazy CORPORA wykorzystywano także $Fp = 16kHz$. W podrozdz. 4.6.6 i 4.6.7 wykonywano także testy z wykorzystaniem stałej liczby próbek. Liczba bitów wynosiła 12 dla testów wykonanych z użyciem bazy CORPORA i 16 bitów dla danych z WBI. We wszystkich testach wykorzystano sygnał monofoniczny.

Na podstawie wniosków z badań z podrozdz. 4.5.4 zdecydowano o zastosowaniu filtru preemfazy o transmitancji $1 - 0.95z^{-1}$ przed algorytmem wyboru początku i końca słowa.

W testach, w których wykorzystywano algorytm wyboru początku i końca słowa, korzystano standardowo z algorytmu VAD-1SVM o wartości początkowej współczynnika $v = 0{,}15$, jeśli nie zaznaczono inaczej. W kilku eksperymentach wykorzystano do badań także algorytm MRSED, głównie w celach porównawczych. Szczegółowo budowę i parametry obu algorytmów przedstawiono w podrozdz. 4.5.

Po skróceniu sygnału o te fragmenty, które nie zostały przez algorytm VAD zaklasyfikowane jako mowa, następuje proces dzielenia sygnału na ramki o długości 240 próbek dla $Fp = 8kHz$ (300 dla 16kHz) i skoku równemu 80 próbek przy $Fp = 8kHz$ (100 dla 16kHz) zgodnie z zależnością (2.7). Następnie dane z każdej ramki poddano procesowi okienkowania wykorzystując okno Hamminga zgodnie z (2.17).

Kolejnym etapem jest parametryzacja sygnału mowy zawartego w ramkach. Nie wiadomo z góry, który z parametrów okaże się najskuteczniejszy, dlatego zdecydowano się na przetestowanie w sumie ośmiu typów parametrów. Większość testów wykonano z użyciem współczynników BFCC (ang. *Bark Frequency Cepstrum Coefficients*). Do pozostałych badanych parametrów należą także:

- współczynniki liniowego kodowania predykcyjnego cepstrum (LPCC),
- współczynniki cepstrum melowego (MFCC)
- współczynniki liniowego kodowania predykcyjnego (LPC),
- współczynniki odbicia (RC),
- współczynniki logarytmicznego przekroju tuby akustycznej (LAR),

- częstotliwości widma liniowego (LSF),
- współczynniki odwrotnej funkcji sinus (ISP).

Dokładny opis każdego z tych parametrów można znaleźć w podrozdz. 2.2.

Liczbę parametrów, które obliczano z każdej z ramek, ustalono na 10. Tylko w teście w podrozdz. 4.6.15 zwiększono liczbę parametrów do 12.

Wszędzie tam, gdzie wykorzystywano współczynniki liniowego kodowania predykcyjnego, do ich obliczenia korzystano z metody autokorelacyjnej wykorzystującej algorytm Levinsona-Durbina.

W przypadku parametrów cepstralnych BFCC i MFCC wykorzystano filtry trójkątne o częstotliwości dolnej 0Hz i górnej 4kHz przy liczbie pasm spektralnych (ang. *warped spectra bands*) wynoszącej 40 i współczynniku eksponenty w procesie lifteringu równym 0.6.

W większości testów stosowano technikę uzupełnienia zerami do najdłuższego z wektorów podawanych na wejście sieci SVM. Testowano także metodę skracania do najkrótszego z wektorów oraz stałą liczbę próbek w wektorze (patrz. podrozdz. 4.6.6 i 4.6.7).

Do procesu klasyfikacji wykorzystano sieci typu C-SVC (ang. *C-Support Vector Classification*) wykorzystując w testach głównie jądro radialne. W podrozdz. 4.6.10 testowane są także jądra: liniowe, wielomianowe i sigmoidalne. Ze względu na fakt, że rozpoznawanych jest wiele klas, należało wybrać z pomiędzy algorytmów wielokrotnej klasyfikacji (patrz podrozdz. 3.7). Zdecydowano się na wybór algorytmu „*jeden przeciw jednemu*" zgodnie z [87]. W większości testów stosowane jest skalowanie wektorów parametrów w zakresie od -1 do 1. Stosowana jest także 5-stopniowa metoda *cross-validation* w połączeniu z metodą *grid-search* poszukiwania optymalnej pary parametrów C i γ. W niektórych testach całkowicie zrezygnowano z tej metody lub stosowano 9-stopniową metodę *cross-validation*. Jeśli nie korzystano z metody *cross-validation grid-search* (CVGS), to domyślną wartością dla parametru C byłą wartość 1 a współczynnik jądra γ był równy odwrotności długości wektorów danych wejściowych.

Warunki pomiarów i uwagi do testów:

W znacznej większości eksperymentów testy odbywały się na innych sygnałach mowy niż te, które zostały użyte w procesie uczenia. Jeśli było inaczej, zostało to zaznaczone w uwagach do testów.

Jeśli zastosowano metodę *cross-validation*, to w takim przypadku także stosowano metodę *grid-search* poszukiwania optymalnej pary parametrów γ i C. Jeśli nie zastosowano metody *cross-validation*, to nie stosowano też metody *grid-search*.

Przy obliczaniu parametrów MFCC, BFCC i LPCC korzystano z [114]. Do obliczania pozostałych metod parametryzacji korzystano z m-plików dostępnych w Matlabie.

Najczęściej tematem rozpoznawania były *imiona*. W przypadku testów z bazą WBI zawsze, natomiast w przypadku bazy CORPORA, w niektórych testach stosowane były także cyfry od *„zero"* do *„dziewięć"* lub 33 litery alfabetu.

W większości testów sieć podejmowała decyzję o klasyfikacji, dokonując wyboru spośród 20 klas. W niektórych przypadkach stosowano także testy 10 lub 15 klas a w przypadku testów, gdzie dokonywano klasyfikacji liter alfabetu, liczba klas wynosiła 33.

Eksperymenty w środowisku zaszumionym dokonywano w zależności od typu testów przy stosunku sygnału do szumu na poziomie: szumu naturalnego[14], 45dB, 28dB, 24dB, 20dB, 16dB, 12dB, 8dB, a nawet 4dB.

W przypadku sygnałów mowy z bazy CORPORA, należało je w przypadku eksperymentów, gdzie stosowano algorytm VAD, dodatkowo wydłużyć w sposób sztuczny. Aby uniknąć wykrywania przez algorytmy wyboru początku i końca słowa miejsc połączenia oryginalnego sygnału z próbkami o wartości zero, postanowiono zatrzeć ślad wydłużenia poprzez dodanie do sygnału wydłużonego sygnału zawierającego szum różowy o $SNR = 45dB$ w stosunku do danego sygnału mowy.

4.6.3 Pomiar skuteczności rozpoznawania i poziom pewności pomiarów

Skuteczność rozpoznawania

Skuteczność rozpoznawania oceniano zgodnie z

$$WR = 1 - e_r = 1 - \frac{n_b}{T} = \frac{n_p}{T} 100\% \tag{4.36}$$

gdzie:

WR - skuteczność rozpoznawania w procentach,

e_r - stopa błędu,

n_b - liczba błędnie rozpoznanych słów w czasie testów,

n_p - liczba poprawnie rozpoznanych słów w czasie testów,

T - długość ciągu testowego, czyli liczba rozpoznawań (określana dalej także jako $Testy\ [liczba\ plików]$).

Każdy błąd oznacza błędne przyporządkowanie do jednej z klas. W testach zakłada się, że każde słowo poddawane rozpoznawaniu należy do jednej z klas, do których należały dane uczące.

Pewność pomiarów

Można zadać pytanie, z jaką pewnością otrzymane wyniki zgodne są z rzeczywistością? Statystyczny błąd pomiaru z 95% pewnością można obliczyć zgodnie z [77 p. 262], używając wzoru

$$\frac{\Delta}{2} = 1{,}96\sqrt{\frac{p(100-p)}{n}}, \tag{4.37}$$

[14] Określenie „szum naturalny” wyjaśniono w uwagach w podrozdz. 4.4.2.

gdzie: p – współczynnik rozpoznawania słów oraz n – całkowita liczba wypowiedzi poddawanych rozpoznawaniu. Dzięki temu otrzymane wyniki należą do przedziału $[p-\frac{\Delta}{2}, p+\frac{\Delta}{2}]$ z prawdopodobieństwem 95%.

Początkowo rozważano możliwość zamieszczenia przedziałów ufności w tabelach i wykresach. Ostatecznie jednak zrezygnowano z tego pomysłu, aby nie zaciemniać wyników pomiarów.

Najczęściej w testach z wykorzystaniem bazy WBI stosowano całkowitą liczbę wypowiedzi poddawanych rozpoznawaniu równą 200. Zakładając, że średni współczynnik rozpoznawania *p* wynosił np. 90%, to statystyczny błąd pomiaru z 95% pewnością będzie wynosić 1,31% zgodnie z (4.37).

4.6.4 Wpływ wyboru metody parametryzacji na wyniki rozpoznawania w przypadku wielu mówców i braku algorytmu VAD

Celem eksperymentu jest porównanie wpływu różnych metod parametryzacji na wyniki rozpoznawania przy założeniu, że z systemu będzie korzystało wielu mówców. Ze względu na fakt, że algorytm wyboru początku i końca słowa nie jest związany w sposób bezpośredni z badanymi parametrami, postanowiono uczyć i testować na sygnałach z bazy CORPORA posiadających oznaczony już początek i koniec słowa. W testach przedstawiono także wpływ parametrów C i γ na skuteczność rozpoznawania. Wyniki eksperymentów przedstawiono w Tab. 4.5 i Tab. 4.6.

Warunki testu:

Baza danych	**CORPORA**
Etykiety	Imiona
Liczba klas	20
Uczenie [liczba plików]	420
Testy [liczba plików]	400
Rozdz. pomiaru	0,25%
Algorytm VAD	Brak
k	1440

Testy dla każdej z badanych metod parametryzacji dokonano dwukrotnie. W pierwszym teście stosowano metodę *cross-validation* (5 lub 9 stopniową) wraz z metodą *grid-search* znajdywania optymalnej pary parametrów C i γ a w drugim przypadku stosowano domyślne wartości $C = 1$ oraz $\gamma = 1/k$.

Tab. 4.5 Porównanie wpływu parametrów cepstralnych: BFCC, MFCC i LPCC na wyniki rozpoznawania (wielu mówców, brak algorytmu VAD)

Parametr	BFCC	BFCC*	MFCC	MFCC	MFCC*	LPCC	LPCC*
log2(γ)	-13	γ=1/1440	-13	-9	γ=1/1440	-7	γ=1/1440
log2(C)	7	0	7	3	0	1	0
LDCV	5	-	5	**9**	-	5	-
SV	**392**	420	399	405	420	418	420
CVGS [%]	**94,3**	82,1	94,0	**95,0**	83,1	**92,9**	85,2
WR(SN) [%]	**93,8**	82,0	**95,3**	94,5	83,0	12,5	**86,0**
WR(20dB) [%]	**93,8**	76,8	**92,5**	89,8	78,0	11,0	**79,8**
WR(12dB) [%]	**86,5**	71,8	**81,3**	71,5	68,8	9,8	**59,8**
WR(4dB) [%]	**69,5**	58,8	49,8	39,8	**52,8**	9,8	**32,8**

* oznacza, że ustawiono domyślne wartości: C = 1 oraz γ = *1/k = 6,944E-04*.

Tab. 4.6 Porównanie wpływu pochodnych parametrów LPC na wyniki rozpoznawania (wielu mówców, brak algorytmu VAD)

Parametr	LPC	LPC*	RC	RC*	LAR	LAR*	LSF	LSF*	ISP	ISP*
log2(γ)	-9	γ=1/1440	-9	γ=1/1440	-9	γ=1/1440	**-13**	γ=1/1440	-9	γ=1/1440
log2(C)	3	0	3	0	3	0	**9**	0	3	0
LDCV	5	-	5	-	5	-	**9**	-	5	-
SV	407	420	417	420	417	420	**402**	420	417	420
CVGS [%]	**86,2**	60,0	**86,4**	72,9	**85,0**	68,1	**86,7**	39,3	**86,4**	70,0
WR(SN) [%]	40,0	**68,5**	56,0	**77,8**	**81,0**	74,5	**90,0**	48,0	48,8	**76,8**
WR(20dB) [%]	24,5	**49,8**	53,0	**68,3**	**71,3**	65,3	**77,0**	40,0	43,0	**68,8**
WR(12dB) [%]	18,0	**43,3**	22,0	**55,5**	55,8	**56,8**	**54,0**	32,3	17,0	**55,0**
WR(4dB) [%]	17,3	**32,0**	7,8	**33,3**	33,5	**37,5**	**29,5**	23,8	8,5	**36,3**

* oznacza, że ustawiono domyślne wartości: C = 1 oraz γ = 1/k = 6,944E-04.

Wnioski z eksperymentu:

W przypadku badań, w których uczono i testowano sieć SVM na sygnałach pochodzących od różnych mówców widać, że dla niektórych metod parametryzacji (LPCC, LPC, RC czy ISP) zastosowanie metody *cross-validation* i *grid-search* nie przyniosło zakładanego poziomu rozpoznawania (patrz parametr CVGS[%] i WR(SN)[%] w Tab. 4.5 i Tab. 4.6). Wartości liczbowe tych wyników powinny być zbliżone. W związku z tym postanowiono dokonać powtórnych testów z domyślnymi ustawieniami $C = 1$ oraz *γ = 1/k = 6,944E-04*.

Porównując otrzymane wyniki wcześniejszych i późniejszych badań widać wyraźnie, że zastosowanie standardowych ustawień parametrów C i γ poprawiło wyniki rozpoznawania metod LPCC, LPC, RC czy ISP. Jednocześnie jednak wpłynęło to negatywnie na skuteczność takich parametrów jak BFCC, MFCC czy LSF.

Wyniki oprócz tego, że pokazują różnice w skuteczności działania różnych metod parametryzacji, uwidaczniają, jak duży wpływ na wynik rozpoznawania ma odpowiedni dobór pary parametrów C i γ.

Warto zauważyć, że tam, gdzie otrzymano najlepsze wyniki, γ miała najmniejszą wartość a C największą (128 dla BFCC i MFCC oraz 512 dla parametrów LSF).

Warto także zwrócić uwagę, że praktycznie dla każdej z metod występuje stosunkowo duża liczba wektorów podtrzymujących, stanowiących od 93,3 do 100% wszystkich wektorów uczących.

Jedynie parametry BFCC (93,8%), MFCC (95,3%) i LSF (90%) uzyskały skuteczność rozpoznawania na poziomie równym lub wyższym od 90% przy testach na danych pozbawionych zewnętrznego szumu.

Po zastosowaniu szumu na poziomie SNR = 20dB jedynie parametry BFCC (93,8%) i MFCC (92,5%) utrzymały skuteczność powyżej 90%. Pozostałe parametry nie przekroczyły 80% skuteczności.

Po zwiększeniu szumu do SNR = 12dB ukazują się wyraźne różnice w skuteczności pomiędzy parametrami BFCC (86,5%) i MFCC (81,3%) a pozostałymi nieprzekraczającymi 60% skuteczności.

W przypadku dużego zaszumienia (SNR = 4dB) bezkonkurencyjnie wypadła metoda BFCC (69,5%) pokonując MFCC (52,8%) i pozostawiając daleko w tyle pozostałe metody, których skuteczność nie przekroczyła w najlepszym wypadku 40%.

Ze względu na fakt, że uzyskane wyniki tak bardzo są zróżnicowane ze względu na parametry C i γ, postanowiono dokonać kolejnych testów porównujących skuteczność różnych metod parametryzacji, tym razem z wykorzystaniem bazy imion WBI, nagrywanej przez jednego mówcę.

4.6.5 Wpływ wyboru metody parametryzacji na wyniki rozpoznawania w przypadku jednego mówcy i zastosowania algorytmu VAD

Celem badania jest porównanie wpływu różnych metod parametryzacji na wyniki rozpoznawania przy założeniu, że z systemu będzie korzystał jeden użytkownik. Jest to badanie dodatkowe, mające zweryfikować wyniki uzyskane w podrozdziale 4.6.4. Dodatkowo, aby uwiarygodnić wyniki rozpoznawania, wprowadzono algorytm wyboru początku i końca słowa. Wyniki eksperymentów przedstawiono w Tab. 4.7 i Tab. 4.8.

Warunki testu:

Baza danych	WBI
Etykiety	Imiona
Liczba klas	20
Uczenie [liczba plików]	200
Testy [liczba plików]	200
Rozdz. pomiaru	0,5%
Algorytm VAD	**VAD-1SVM** ($v = 0{,}15$)

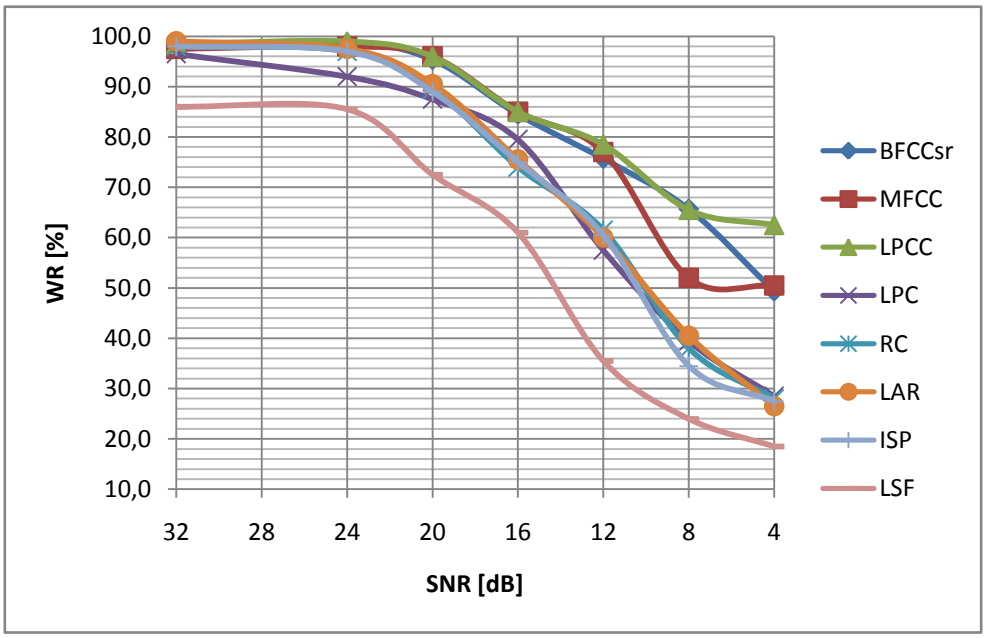
Testy parametru BFCC powtórzono trzykrotnie, a następnie ich wyniki uśredniono (BFCCsr), zamieszczając je na wykresie oraz posługując się danymi uśrednionymi dla tego parametru w analizie wyników badań.

Tab. 4.7 Porównanie wpływu parametrów cepstralnych: BFCC, MFCC i LPCC na wyniki rozpoznawania (jeden mówca, algorytm VAD)

Parametr	BFCC(1)	BFCC(2)	BFCC(3)	BFCCsr	MFCC	LPCC
log2(γ)	-13	-13	-13	-13	2	-13
log2(C)	7	7	7	7	-5	7
SV	194	194	194	194	200	196
CVGS [%]	97,5	97,5	97,5	97,5	97,5	**98,5**
WR(SN) [%]	97,5	97,5	97,5	97,5	97,5	**98,5**
WR(24dB) [%]	98,0	**99,0**	**99,0**	98,7	98,0	**99,0**
WR(20dB) [%]	**96,0**	95,0	95,0	95,3	**96,0**	**96,0**
WR(16dB) [%]	**86,0**	83,5	84,0	84,5	85,0	85,0
WR(12dB) [%]	76,5	74,0	76,5	75,7	77,0	**78,5**
WR(8dB) [%]	**67,5**	64,0	65,5	65,7	52,0	65,5
WR(4dB) [%]	50,5	45,0	52,5	49,3	50,5	**62,5**

Tab. 4.8 Porównanie wpływu pochodnych parametrów LPC na wyniki rozpoznawania (jeden mówca, algorytm VAD)

Parametr	LPC	RC	LAR	ISP	LSF
log2(γ)	-13	-11	-13	-13	-15
log2(C)	9	5	7	7	9
SV	178	196	195	196	198
CVGS [%]	97,0	97,5	97,0	97,0	90,0
WR(SN) [%]	96,5	98,0	**99,0**	98,0	86,0
WR(24dB) [%]	92,0	97,0	97,5	97,0	85,5
WR(20dB) [%]	87,5	89,5	90,5	89,0	72,5
WR(16dB) [%]	79,5	74,0	75,5	75,0	61,0
WR(12dB) [%]	57,5	61,5	60,0	60,5	35,5
WR(8dB) [%]	39,5	38,0	40,5	34,5	24,0
WR(4dB) [%]	28,5	28,0	26,5	27,5	18,5



Rys. 4.21 Wykres skuteczności rozpoznawania systemu dla różnych metod parametryzacji przy różnym SNR (jeden mówca, algorytm VAD)

Wnioski z eksperymentu:

Na Rys. 4.21 widać wyraźnie, że parametry cepstralne charakteryzują się wyższą skutecznością rozpoznawania niż alternatywne formy reprezentacji współczynników LPC. Najsłabiej w ogólnym rozrachunku wypadły parametry LSF. Różnica w skuteczności rozpoznawania jest tym większa, im większy jest szum, któremu poddane były dane testowe.

Praktycznie przy szumie nie większym niż 24dB wszystkie typy parametrów uzyskiwały od 96,5% do 99%, poza nieco słabszymi parametrami LPC (92% przy SNR = 24dB) oraz wyraźnie odbiegającymi od pozostałych parametrami LSF (85,5% przy SNR = 24dB).

Parametry cepstralne: MFCC, BFCC i LPCC przy SNR ≥ 12dB włącznie uzyskiwały bardzo zbliżone wyniki skuteczności rozpoznawania, na poziomie powyżej 75%, w przeciwieństwie do pozostałych metod pokrewnych LPC (LPC, LAR, ISP, RC), mających skuteczność na poziomie ok. 60% oraz wyraźnie słabszej metodzie LSF (35,5%).

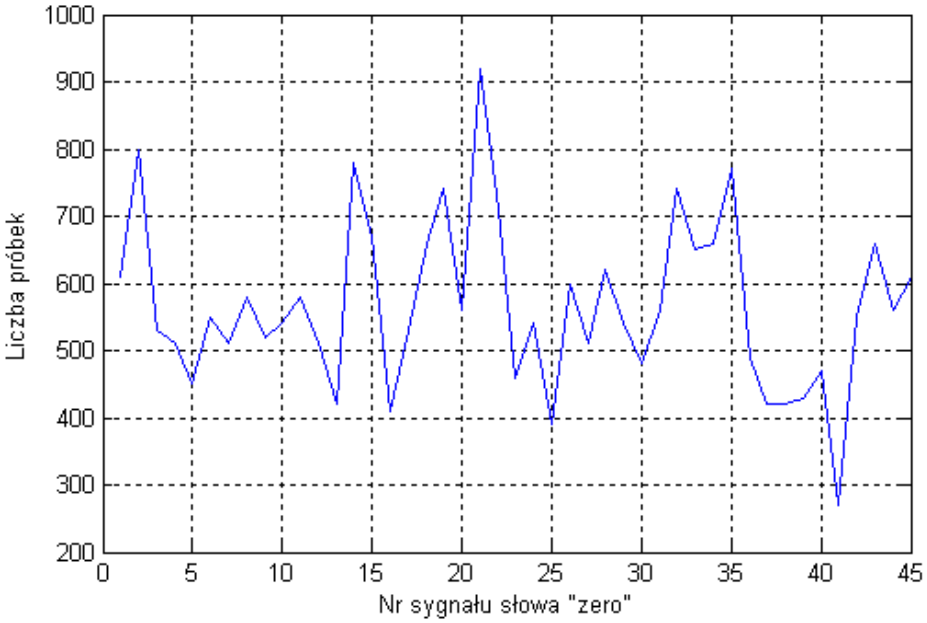
Po zwiększeniu szumu do SNR = 8dB najlepiej wciąż zachowywały się parametry BFCC i LPCC utrzymując współczynnik rozpoznawania na poziomie ponad 65%. Największą skuteczność przy SNR = 4dB uzyskały w teście parametry LPCC (62,5%) a najmniejszą parametry LSF (18,5%).

Można także zauważyć, że zastosowanie metody *cross-validation* w połączeniu z *grid-search* przyniosło zamierzony efekt, uzyskując podobne wartości liczbowe pomiędzy CVGS a WR(SN) z wysokim współczynnikiem rozpoznawania.

4.6.6 Wpływ długości wektorów na wyniki rozpoznawania

Celem eksperymentu jest zbadanie wpływu długości wektorów parametrów, na których sieć SVM jest uczona i testowana. Podobnie, jak w przypadku klasycznych sieci neuronowych istnieje konieczność, aby wektory wejściowe miały równą długość.

Zdecydowano o przeprowadzeniu kilku typów testów. W pierwszym przypadku wszystkie wektory skracano do najkrótszego ze wszystkich testowanych wektorów, który liczył zaledwie 160 parametrów. Taką ich liczbą było reprezentowane jedno z powtórzeń słowa „*trzy*”. W drugim teście wszystkie wektory parametrów uzupełniano zerami do najdłuższego z wektorów liczącego 1290 parametrów. Taką liczbą parametrów było reprezentowane jedno z powtórzeń słowa „*dziewięć*”.



Rys. 4.22 Różna długość tego samego słowa *„zero”* nagrana przez różne osoby przy *Fp = 8kHz*

Ostatnie podejście do problemu stałej długości wektorów jest zupełnie odmienne od wcześniejszych. Na Rys. 4.22 widać, że to samo słowo „*zero*” wypowiedziane przez różne osoby można zapisać, przy tej samej częstotliwości próbkowania 8kHz, z użyciem poniżej 300 lub powyżej 900 próbek. To samo słowo więc, wypowiedziane przez różne osoby w sposób naturalny, może zostać wypowiedziane trzykrotnie szybciej (lub wolniej).

Założono, że być może rozwiązaniem jest nie tyle skracanie lub wydłużanie sygnałów, lecz taka normalizacja sygnałów, aby każdy z nich zawierał stałą liczbę próbek. Dzięki temu teoretycznie powinien zniknąć problem rozpoznawania nie tylko tych samych słów, które zostały wypowiedziane z różną długością, lecz również różnych słów dzięki założeniu, że każda wypowiedź z góry będzie miała stałą, równą liczbę próbek.

Wykorzystując funkcję „*resample*” ze środowiska Matlaba, dokonano przepróbkowywania sygnałów ze stałej częstotliwości do stałej ilości próbek wynoszącej 5000. Funkcja ta wykorzystuje antyaliasingowy dolnoprzepustowy filtr o skończonej odpowiedzi impulsowej w procesie resamplingu kompensując opóźnienie filtru. Filtr ten wykorzystuje okno Kaisera.

Wyniki eksperymentów przedstawiono w Tab. 4.9.

Warunki testu:

Baza danych	CORPORA
Etykiety	**cyfry (0-9)**
Liczba klas	10
Uczenie [liczba plików]	450
Testy [liczba plików]	450
Rozdz. pomiaru	0,22%
Algorytm VAD	**VAD-1SVM** ($v = 0{,}05$)

Jest to jedyny test, w którym uczenie i testowanie odbywa się na tych samych danych.

Tab. 4.9 Wpływ długości wektorów na wyniki rozpoznawania

Oznaczenie testu	**A**	**B**	**C**	**D**	**E**	**F**	**G**
Opis	**skracanie do najkrótszego z wektorów (160 parametrów)**	**uzupełnianie zerami do najdłuższego z wektorów (1290 parametrów)**	**stała liczba próbek**	**uzupełnianie zerami do najdłuższego z wektorów (1290 parametrów)**	**stała liczba próbek**	**stała liczba próbek**	**uzupełnianie zerami do najdłuższego z wektorów (k = 1290 parametrów), algorytm MRSED zamiast VAD-1SVM**
SNR[dB] - testy	45	45	45	20	20	20	20
SNR[dB] - uczenie	45	45	45	45	45	SN	SN
Fp [kHz]	8	8	-	8	-	-	8
Liczba próbek	-	-	5000	-	5000	5000	-
log2(γ)	-5	-11	-13	-11	1	-11	-13
log2(C)	7	5	9	5	-5	5	7
CVGS [%]	16,4	95,8	95,8	94,7	98,0	97,6	97,3
WR [%]	**16,0**	**99,8**	**100,0**	**88,4**	**87,8**	**77,1**	**48,0**

Wnioski z eksperymentu:

Skracanie do najkrótszego z parametrów (A) okazało się w ogóle nieskuteczne. Mimo, że technika ta jest najszybsza w działaniu, bo ilość przetwarzanych parametrów redukowana jest do minimum (w tym wypadku ponad 8-krotnie), to otrzymano skuteczność rozpoznawania jedynie na poziomie 16% przy minimalnym szumie porównywalnym z szumem tła (SNR = 45dB). Zdecydowano więc, że opcja ta zostaje odrzucona i nie będzie rozważana w dalszych testach.

Wyniki badań metodą uzupełniania zerami do najdłuższego z wektorów w porównaniu z metodą stałej liczby próbek okazały się zbliżone uzyskując skuteczność rozpoznawania bliską 100% (B i C) przy SNR = 45dB oraz ok. 88% (D i E) przy SNR = 20dB.

Kolejny test (F) ma charakter „roboczy” potwierdzając, że zasadne było uczenie po dodaniu szumu na poziomie 45dB. Ze względu na fakt wydłużania sygnałów w bazie CORPORA, o którym wspominano już wcześniej, niedodanie szumu do wydłużonego sygnału powoduje pogorszenie wyników rozpoznawania o ponad 10% w stosunku do testu E.

W teście E zastosowano algorytm MRSED wyboru początku i końca i uczono przy naturalnym szumie osiągając jedynie 48% skuteczności. Jest to o ponad 40% niższa skuteczność rozpoznawania w porównaniu z testem D, w którym zastosowano algorytm VAD-1SVM i dodano uczenie przy SNR = 45dB osiągając poziom skuteczności rozpoznawania powyżej 88% przy SNR = 20dB.

Ze względu na to, że przy uzupełnianiu zerami do najdłuższego z parametrów oraz normalizacji względem stałej liczby próbek osiągnięto podobne wyniki (testy B i C przy SNR = 45dB) oraz (D i F dla SNR = 20dB), zdecydowano o dokonaniu dodatkowego testu mającego zdecydować o wyborze jednej z metod do tworzonego systemu rozpoznawania mowy.

4.6.7 Porównanie metody uzupełniania wektorów zerami z metodą stałej liczby próbek

Na podstawie wyników eksperymentów przeprowadzonych w podrozdz. 4.6.6 uznano metodę skracania do najkrótszego z wektorów za nieskuteczną, rezygnując z niej całkowicie w opracowywanym systemie rozpoznawania izolowanych słów.

Ze względu na fakt, że metoda uzupełnienia zerami do najdłuższego z wektorów oraz normalizacja do stałej liczby próbek uzyskały zbliżone wyniki, zdecydowano o dokonaniu dokładnego testu porównawczego obu metod przy różnych SNR.

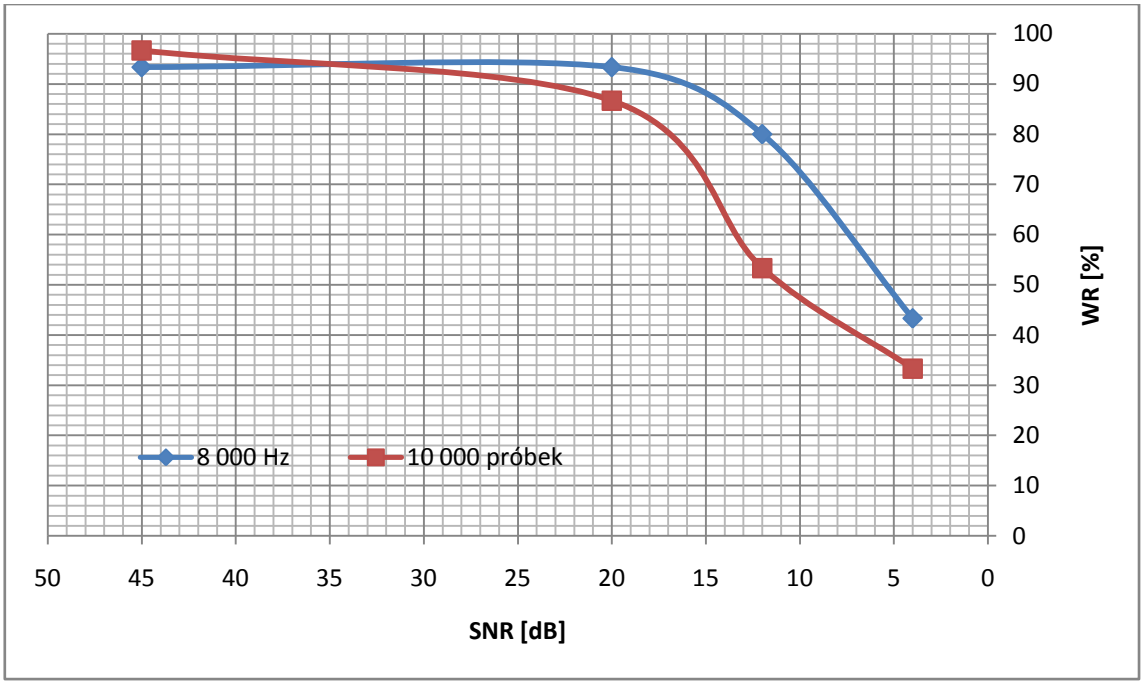
Warunki testu:

Baza danych	CORPORA
Etykiety	Cyfry (0-9)
Liczba klas	10
Uczenie [liczba plików]	250
Testy [liczba plików]	30
Rozdz. pomiaru	3,33%
Algorytm VAD	VAD-1SVM ($\nu = 0{,}05$)

Zawężono przy tym grupę mówców. Uczenie i testy dokonywane były tylko na nagraniach głosów męskich z przedziału wiekowego od 20 do 70 lat. Proces uczenia odbywał się przy SNR = 45dB.

Tab. 4.10 Porównanie metody uzupełniania wektorów zerami z metodą stałej liczby próbek przy różnym SNR

Opis	**uzupełnianie zerami do najdłuższego z wektorów (1290 parametrów)**	**stała liczba próbek**
Fp [kHz]	**8**	-
Liczba próbek	-	**10 000**
log2(γ)	-13	-15
log2(C)	7	7
CVGS [%]	95	93
WR(45dB) [%]	93	97
WR(20dB) [%]	93	87
WR(12dB) [%]	80	53
WR(4dB) [%]	43	33



Rys. 4.23 Wykres porównujący metodę uzupełniania wektorów zerami z metodą stałej liczby próbek przy różnym SNR

Wnioski z eksperymentu:

W Tab. 4.10 i na Rys. 4.23 przedstawiono wyniki eksperymentu.

Analizując wyniki testów widać wyraźnie, że metoda stałej liczby próbek osiągnęła o 4% lepszą skuteczność rozpoznawania jedynie przy SNR = 45dB czyli w sytuacji, gdy poziom szumu jest minimalny. W pozostałych trzech testach uzyskano lepsze wyniki przy uzupełnianiu zerami do najdłuższego z wektorów. Otrzymane wyniki były lepsze odpowiednio o 6% przy 45dB, aż o 27% przy 12dB oraz o 10% przy 4dB w porównaniu z metodą stałej liczby próbek.

W sposób świadomy zwiększono stałą liczbę próbek do 10 000, mając na celu poprawę jakości rozpoznawania. Mimo tego zabiegu wyniki skuteczności rozpoznawania przemawiają wyraźnie na korzyść metody uzupełniania wektorów zerami do najdłuższego z wektorów.

Zastosowanie stałej liczby próbek przy większej ich ilości bardzo wydłuża cały proces rozpoznawania. Biorąc pod uwagę także kryterium szybkości obliczeń, nie pozostaje wątpliwości, że metoda stałej liczby próbek, mimo że teoretycznie ciekawa, nie nadaje się do zastosowania w projektowanym systemie rozpoznawania mowy, ustępując miejsca tradycyjnej metodzie uzupełniania zerami do najdłuższego z wektorów.

4.6.8 Wpływ wartości *C* i *γ* na wyniki rozpoznawania oraz liczbę wektorów podtrzymujących

Celem eksperymentów jest zbadanie, jaki wpływ ma zmiana parametru C albo γ na skuteczność rozpoznawania oraz na liczbę wektorów podtrzymujących.

Warunki testu:

Baza danych	WBI
Etykiety	Imiona
Liczba klas	20
Uczenie [liczba plików]	200
Testy [liczba plików]	200
Rozdz. pomiaru	0,5%
Algorytm VAD	VAD-1SVM ($v = 0{,}15$)
k	990
SNR[dB]	Szum naturalny

Wartość $\gamma = \frac{1}{k}$ jest wartością domyślną stosowaną np. wtedy, gdy nie korzystano z metody *grid-search* poszukiwania optymalnej wartości pary C i γ. W Tab. 4.11 zamieszczono wyniki testu zmiany parametru C przy stałym współczynniku γ wynoszącym $\frac{1}{990}$.

Tab. 4.11 Wpływ parametru *C* na skuteczność rozpoznawania i liczbę wektorów podtrzymujących

Liczba SV	WR [%]	log2(*C*)	*C*
200	96	-5	0,03125
200	96	-4	0,0625
200	96	-3	0,125
200	96	-2	0,25
200	96	-1	0,5
200	96	0	1
199	97,5	1	2
197	97,5	2	4
196	97,5	3	8
196	97,5	4	16
196	97,5	5	32
196	97,5	6	64
196	97,5	7	128
196	97,5	8	256
196	97,5	9	512
196	97,5	10	1024
196	97,5	11	2048
196	97,5	12	4096
196	97,5	13	8192
196	97,5	14	16384
196	97,5	15	32768

Tab. 4.12 zawiera wyniki wpływu na skuteczność rozpoznawania i liczbę wektorów podtrzymujących zmiany współczynnika γ przy stałej wartości parametru C. W tym teście, jako stałą wartość parametru C, przyjęto wartość 1.

Tab. 4.12 Wpływ wsp. *γ* na skuteczność rozpoznawania i liczbę wektorów podtrzymujących

Liczba SV	WR [%]	log2(*γ*)	*γ*
200	95,5	-15	3,05E-05
200	95,5	-14	6,10E-05
200	95,5	-13	1,22E-04
200	95,5	-12	2,44E-04
200	95,5	-11	4,88E-04
200	96,0	-10	9,77E-04
200	95,5	-9	1,95E-03
198	96,5	-8	3,91E-03
197	87,5	-7	0,0078125
198	37,0	-6	0,015625
200	15,0	-5	0,03125
200	5,5	-4	0,0625
200	5,0	-3	0,125
200	5,0	-2	0,25
200	5,0	-1	0,5

Wnioski z eksperymentu:

Wyniki zawarte w Tab. 4.11 wskazują, że zmiana parametrów C nie wpływa w sposób znaczący na liczbę wektorów podtrzymujących (SV). Liczba ta jest bardzo duża i zbliżona do całkowitej liczby wszystkich wektorów biorących udział w procesie uczenia wynoszącej 200. Najmniejsza uzyskana liczba SV wynosi 196, co stanowi aż 98% wszystkich wektorów. Oznacza to, że do stworzenia modelu sieci SVM nie zostanie użyte jedynie 2% wektorów biorących udział w uczeniu sieci. W eksperymencie nie zaobserwowano natomiast żadnego wpływu zmiany współczynnika γ na liczbę wektorów podtrzymujących przy stałej wartości parametru C (Tab. 4.12). Może to świadczyć o tym, że liczba wektorów podtrzymujących nie zależy od wartości współczynnika γ.

Przy analizie wpływu parametrów C albo γ na wyniki rozpoznawania widać, że ich wartości mają kluczowy wpływ na uzyskiwane wyniki. W sposób jednoznaczny nie można powiedzieć, że dana wartość tych parametrów jest lepsza od innej. Na podstawie otrzymanych wyników (Tab. 4.12) można jednak zauważyć, że jest pewna tendencja dotycząca wpływu wielkości współczynnika γ na wyniki rozpoznawania. Jeśli wartość współczynnika γ była zbyt duża tzn. w tym przypadku powyżej 2^{-7}, to skuteczność rozpoznawania zaczęła gwałtownie spadać z ponad 96% nawet do 5%.

Ze względu na fakt, że z góry nie wiadomo, jaka wartość parametru C i γ jest optymalna w danym przypadku, zaleca się znajdywanie ich metodą *grid-search*, polegającą na wykonaniu wielu testów z różnymi wartościami par parametrów C i γ. Skuteczność tej metody, w połączeniu z techniką *cross-validation*, zbadano w kolejnym teście w podrozdz. 4.6.9.

4.6.9 Wpływ skalowania i CVGS na wyniki rozpoznawania

Celem testu jest zbadanie wpływu skalowania i metody *cross-validation* w połączeniu z *grid-search* (CVGS) na wyniki skuteczności rozpoznawania.

Warunki testu:

Baza danych	WBI
Etykiety	Imiona
Liczba klas	20
Uczenie [liczba plików]	200
Testy [liczba plików]	200
Rozdz. pomiaru	0,5%
Algorytm VAD	VAD-1SVM
k	990

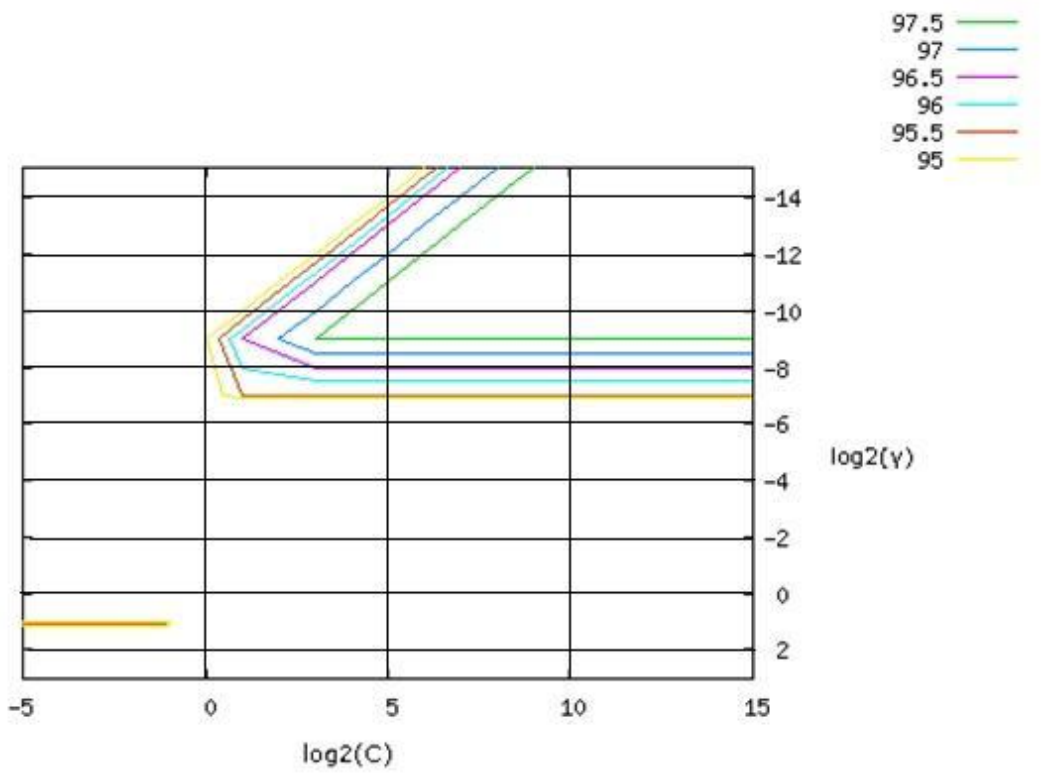
Wyniki testu D w Tab. 4.13 stanowią uśrednione wyniki testów skuteczności rozpoznawania przy różnym SNR i zostały przepisane z Tab. 4.7 (kolumna BFCCsr), aby łatwiej było porównać wyniki.

Tab. 4.13 Wyniki wpływy skalowania i CVGS na wyniki rozpoznawania

Ozn. testu	A	B	C	D
Skalowanie	**Brak**	**Brak**	**[-1,1]**	**[-1,1]**
LDCV	**Brak**	**5**	**Brak**	**5**
log2(γ)	γ = 1/990	-7	γ = 1/990	**-13**
log2(C)	0	-5	0	**7**
CV [%]	-	91,0	-	**97,5**
WR(SN) [%]	**32,5**	**91,5**	96,0	**97,5**
WR(24dB) [%]	29,5	86,5	91,5	**98,7**
WR(20dB) [%]	28,5	79,0	81,0	**95,3**
WR(16dB) [%]	18,5	62,0	67,0	**84,5**
WR(12dB) [%]	11,5	55,0	57,0	**75,7**
WR(8dB) [%]	9,5	37,0	44,0	**65,7**
WR(4dB) [%]	8,0	24,5	31,5	**49,3**

Najsłabsze wyniki uzyskano przy braku skalowania i nie zastosowaniu metody CVGS (test A). Ze względu na niesatysfakcjonujący poziom skuteczności rozpoznawania, rozwiązanie to nie będzie brane pod uwagę w dalszych pracach nad systemem rozpoznawania mowy z użyciem sieci SVM. Zastosowanie metody *cross-validation* w połączeniu z techniką *grid-search*, ale bez procesu skalowania (Test B) znacznie poprawia skuteczność rozpoznawania. Podobnie zastosowanie skalowania bez CVGS (Test C) poprawia jakość rozpoznawania w stosunku do wyników z testu A. Zdecydowanie najlepsze wyniki uzyskano w teście D, stosując skalowanie danych, a następnie metodę *cross-validation*, podziału wektorów uczących na 5 równych części, a następnie poprzez poddanie procesowi znajdywania optymalnej pary parametrów C i γ z wykorzystaniem techniki *grid-search*. W celach testowych zastosowano także podział na 9 równych części wektorów w metodzie *cross-validation*, otrzymując metodą *grid-search* takie same optymalne wartości parametrów C i γ, co w teście D. Wyniki badań z tego podrozdziału uzasadniają, dlaczego standardowo testy w rozdziale wykonywane są z zastosowaniem skalowania oraz większość z nich wykorzystuje także technikę *cross-validation* (5-stopniową) w połączeniu z *grid-search*.

Wadą metody *grid-search* jest stosunkowo długi czas obliczeń, ze względu na fakt wielokrotnego powtarzania procesu uczenia z różnymi parami parametrów C i γ.



Rys. 4.24 Wykres wpływu pary parametrów *C* i *γ* dobieranych metodą *grid-search* na wyniki uczenia sieci SVM

Na Rys. 4.24 przedstawiono wynik zastosowania procesu *grid-search* otrzymany z użyciem biblioteki LIBSVM [84]. Na osi poziomej, w zakresie od -5 do 15 oznaczono parametr x, na którego podstawie można obliczyć C zgodnie z $C = 2^x$. Natomiast na osi pionowej w zakresie od 1 do 15 oznaczono parametr y, na którego podstawie można obliczyć γ zgodnie z $\gamma = 2^y$. Widać wyraźnie, że najlepsze wyniki uczenia sieci uzyskuje się tylko w pewnym zakresie parametrów, w tym wypadku związanym z prawą górną częścią wykresu. Autor uważa, że metoda *grid-search* poszukiwania optymalnej pary C i γ jest na tyle skuteczna, że rekomenduje ją do zastosowania w projektowanym systemie rozpoznawania izolowanych słów.

4.6.10 Wpływ wyboru typu jądra w sieci SVM na wyniki rozpoznawania

Celem testu jest znalezienie najlepszego typu jądra do sieci C-SVC. Testowane będą jądra: wielomianowe, sigmoidalne, liniowe oraz radialne.

Warunki testu:

Baza danych	WBI
Liczba klas	20
Uczenie [liczba plików]	200
Testy [liczba plików]	200
Rozdz. pomiaru	0,5%

LDCV	5
Algorytm VAD	VAD-1SVM

Wykonując badania z użyciem jądra wielomianowego i sigmoidalnego przyjęto, że współczynnik $r = 0$ (patrz Tab. 3.2). Dodatkowo, w przypadku jądra wielomianowego przyjęto stopień wielomianu $p = 3$ (patrz Tab. 3.2).

Wyniki testu z jądrem radialnym stanowią uśrednione wyniki testów skuteczności rozpoznawania przy różnym SNR i zostały przepisane z Tab. 4.7 (kolumna BFCCsr), aby łatwiej było porównać wyniki.

Tab. 4.14 Wpływ typu jądra na wyniki rozpoznawania

Typ jądra	Wielomianowe	Sigmoidalne	Liniowe	Radialne
log2(γ)	-7	-13	-7	-13
log2(C)	-3	7	-5	7
CVGS [%]	96,0	97,5	97,5	97,5
WR(SN) [%]	93,0	97,5	97,5	97,5
WR(20dB) [%]	7,5	95,5	95,5	95,3
WR(16dB) [%]	7,0	83,0	83,5	84,5
WR(12dB) [%]	5,0	73,5	74,0	75,7
WR(8dB) [%]	5,0	62,5	64,0	65,7
WR(4dB) [%]	5,5	44,0	45,0	49,3

Wnioski z eksperymentu:

Najsłabsze wyniki uzyskano dla jądra wielomianowego. Już przy SNR = 20dB skuteczność rozpoznawania była niższa niż 10%, co dyskwalifikuje tę metodę z dalszych badań.

Uzyskano bardzo zbliżone do siebie wyniki pozostałych trzech typów jąder: sigmoidalnego, liniowego i radialnego. Nieznacznie lepsze o kilka procent od pozostałych wyniki uzyskano przy większym zaszumieniu (SNR<12dB) jednak dla jądra radialnego, co skłania do wybrania właśnie tego typu jądra w projektowanym systemie rozpoznawania izolowanych słów z użyciem sieci SVM.

4.6.11 Wpływ wielkości słownika na wyniki rozpoznawania

Spodziewać się należy, że im większa liczba klas, tym bardziej skuteczność rozpoznawania powinna się zmniejszać. Nie wiadomo jednak, jak bardzo. Celem eksperymentu jest zbadanie, jaki wpływ na skuteczność rozpoznawania opracowywanego systemu rozpoznawania izolowanych słów ma liczba klas (w tym wypadku imion).

Warunki testu:

Baza danych	WBI
log2(γ)	-13
log2(*C*)	7
Algorytm VAD	VAD-1SVM

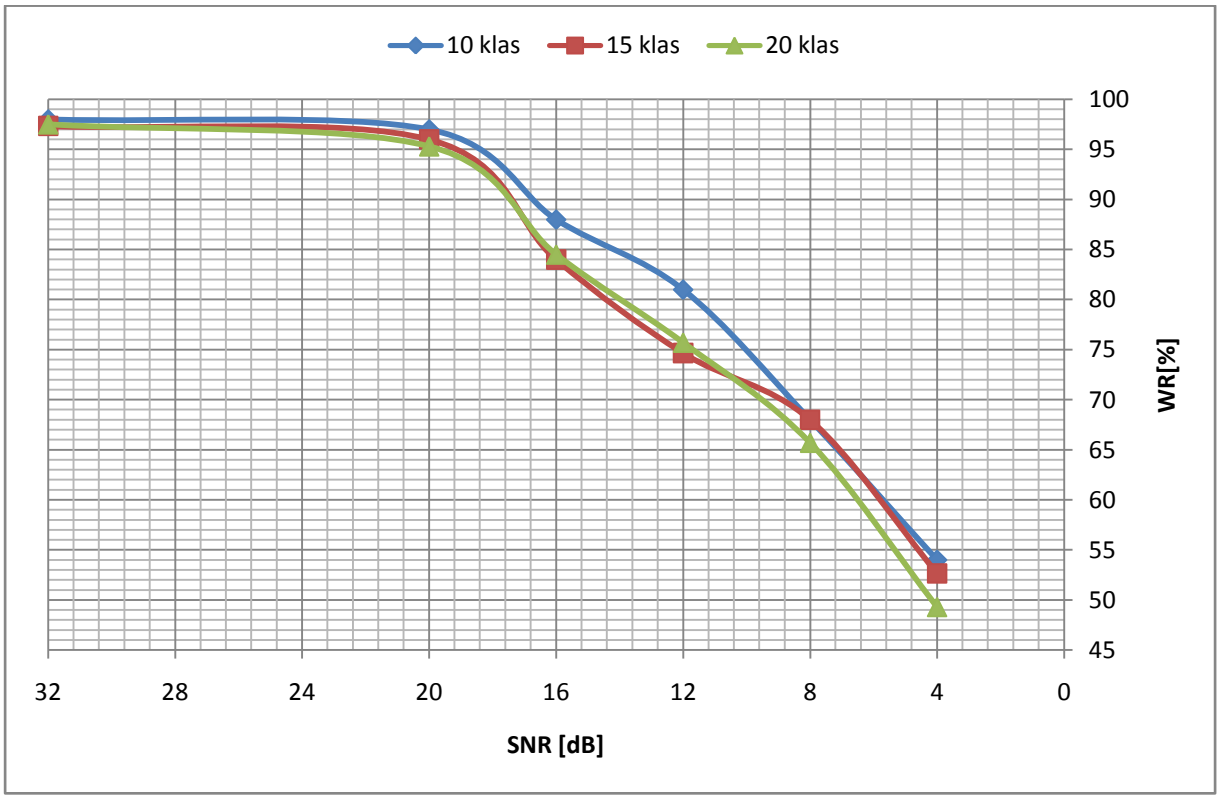
W przypadku, gdy liczba imion wynosiła 10, wykorzystano następujące słowa z WBI: *„Agata”*, *„Barbara”*, *„Czesław”*, *„Dominik”*, *„Ewa”*, *„Feliks”*, *„Grzegorz”*, *„Honorata”*, *„Iwona”*, *„Jarosław”*.

Kiedy liczbę imion zwiększono do 15, uzupełniono słownik o następujących 5 dodatkowych imion: *„Krystyna”*, *„Leszek”*, *„Małgorzata”*, *„Piotr”* i *„Radosław”*.

Wyniki testu dla 20 klas stanowią uśrednione wyniki testów skuteczności rozpoznawania przy różnym SNR i zostały przepisane z Tab. 4.7 (kolumna BFCCsr), aby łatwiej było porównać wyniki. Do testów w tym przypadku wybrano 20 różnych imion, które w porównaniu z listą poprzednich 15 uzupełniono o słowa: *„Sławomir”*, *„Tomasz”*, *„Urszula”*, *„Wioletta”* i *„Zuzanna”*.

Tab. 4.15 Wpływ wielkości słownika na wyniki rozpoznawania

Liczba imion	**10**	**15**	**20**
Uczenie [liczba plików]	100	150	200
Testy [liczba plików]	100	150	200
Rozdz. pomiaru [%]	1	0,67	0,5
CVGS [%]	99,0	98,7	97,5
WR(SN) [%]	98,0	97,3	97,5
WR(20dB) [%]	97,0	96,0	95,3
WR(16dB) [%]	88,0	84,0	84,5
WR(12dB) [%]	81,0	74,7	75,7
WR(8dB) [%]	68,0	68,0	65,7
WR(4dB) [%]	54,0	52,7	49,3



Rys. 4.25 Wykres wpływu liczby klas na skuteczność rozpoznawania przy różnym SNR

Wnioski z eksperymentu:

Zgodnie z oczekiwaniami, generalnie wraz ze wzrostem liczby klas, skuteczność maleje. Nie jest to jednak gwałtowne pogorszenie skuteczności. Największy spadek w porównaniu z najmniejszą liczbą 10 klas wystąpił przy SNR = 12dB i wynosił ponad 6% dla 15 klas oraz ponad 5% w porównaniu z 20 klasami.

Większą różnicę pomiędzy 10 a większą liczbą klas zaobserwowano także przy SNR = 16dB. Różnice w wynikach wynosiły ponad 4% w porównaniu z wynikiem dla 15 klas i o 3,5% w porównaniu z 20 klasami.

Co ciekawe, w trzech testach zaobserwowano nieznacznie lepsze wyniki dla 20 klas w porównaniu z 15 klasami. Różnice te wynosiły odpowiednio 0,2% (SNR = SN), 0,5% (SNR = 16dB) oraz 1% przy SNR = 12dB na korzyść większej liczby klas. Otrzymane wyniki można próbować tłumaczyć tym, że za każdym razem szum różowy jest generowany w sposób losowy.

Analizując uśrednione wyniki dla 20 klas widać, że przy znacznym szumie otrzymuje się wyniki słabsze od pozostałych o ponad 2% przy SNR = 8 dB i o 4,7% (w porównaniu z 10 klasami) i 3,4% (w porównaniu z 15 klasami) przy SNR = 4dB.

Podsumowując cały test należy stwierdzić, że projektowany system rozpoznawania mowy z użyciem sieci SVM z powodzeniem może być stosowany dla 20 klas.

4.6.12 Wpływ liczby powtórzeń na etapie uczenia sieci SVM na wyniki rozpoznawania

Celem testu jest zbadanie, jaki wpływ na skuteczność działania systemu ma liczba powtórzeń każdej z klas, na których odbywa się proces uczenia. Każda klasa była uczona na podstawie odpowiednio 5, 10 i 15 powtórzeń, tej samej etykiety słownej.

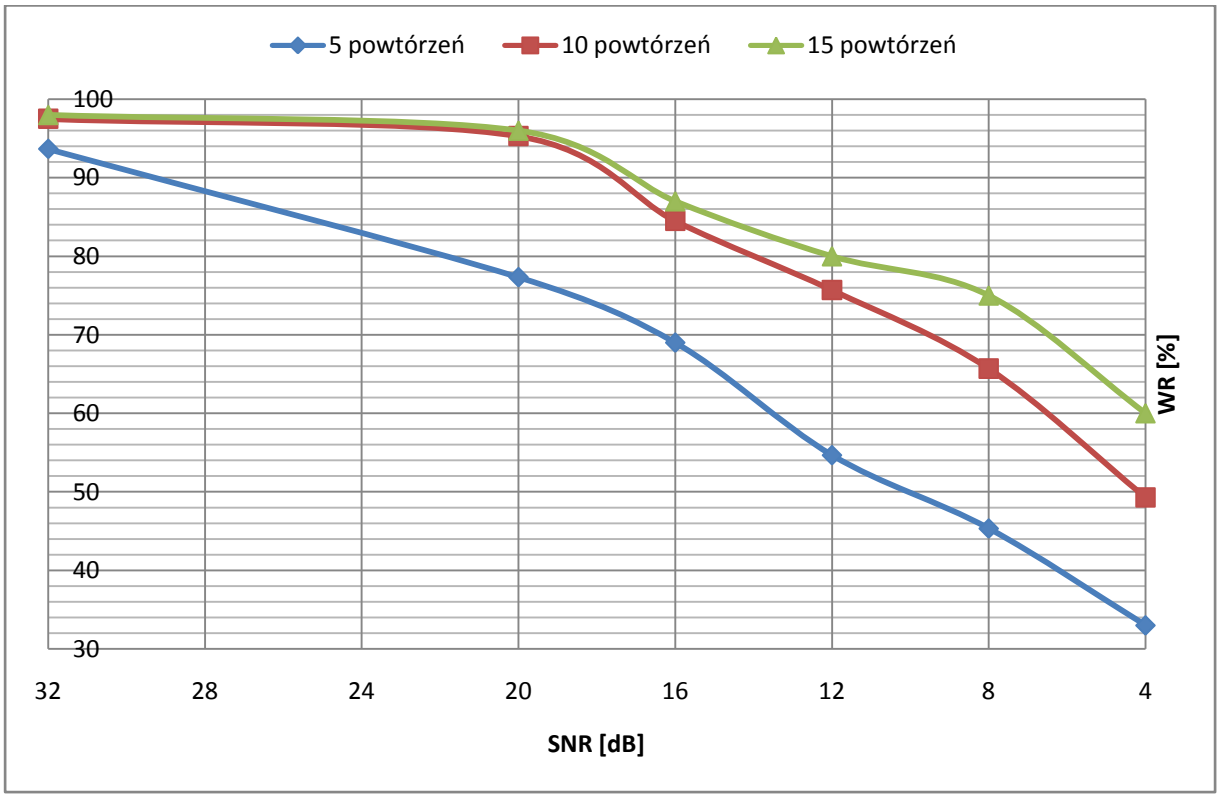
Warunki testu:

Baza danych	WBI
Liczba klas	20
Algorytm VAD	VAD-1SVM

Wyniki testu dla liczby powtórzeń równej 10 stanowią uśrednione wyniki testów skuteczności rozpoznawania, które zostały przepisane z Tab. 4.7 (kolumna BFCCsr), aby łatwiej było porównać wyniki.

Tab. 4.16 Wpływ liczby powtórzeń na etapie uczenia na wyniki rozpoznawania

L. powt. (uczenie)	5	10	15
L. powt. (test)	15	10	5
Uczenie [liczba plików]	100	200	300
Testy [liczba plików]	300	200	100
Rozdz. pomiaru:	0,33%	0,5%	1%
log2(γ)	-13	-13	-13
log2(C)	5	7	7
CVGS [%]	96	97,5	98,0
WR(SN) [%]	93,7	97,5	98,0
WR(20dB) [%]	77,3	95,3	96,0
WR(16dB) [%]	69,0	84,5	87,0
WR(12dB) [%]	54,7	75,7	80,0
WR(8dB) [%]	45,3	65,7	75,0
WR(4dB) [%]	33,0	49,3	60,0



Rys. 4.26 Wykres skuteczności rozpoznawania w zależności od liczby powtórzeń

Wnioski z eksperymentu:

W Tab. 4.16 i na Rys. 4.26 widać wyraźnie, że im większa jest liczba powtórzeń danej klasy na etapie uczenia sieci SVM, tym lepsze są wyniki rozpoznawania. Przy małej liczbie powtórzeń wynoszącej 5, wyniki średnio są gorsze o ok. 20% w porównaniu z liczbą powtórzeń wynoszącą 10. Z kolei najlepsze wyniki uzyskano przy 15 powtórzeniach każdej z klas na etapie uczenia. Różnica w tym wypadku była tym większa, im większe było zaszumienie, wynosząc ponad 10% przy SNR = 4dB.

Może się więc w pierwszej chwili wydawać, że najkorzystniej jest uczyć z jak największą liczbą powtórzeń. Wiąże się to jednak z tym, że model sieci zajmuje więcej pamięci. Ważne jest więc, aby znaleźć kompromis między skutecznością rozpoznawania z jednej strony a prostotą modelu z drugiej. Warto wziąć przy tym pod uwagę stopień zaszumienia środowiska, w którym system będzie pracował, zwiększając odpowiednio liczbę powtórzeń wraz ze wzrostem zaszumienia.

Jeśli z jakiegoś powodu nie dysponuje się wystarczającą liczbą danych uczących, warto wspomóc się wtedy metodą *cross-validation* z większą liczbą podziałów, w celu sztucznego zwiększenia liczby wektorów uczących.

4.6.13 Wpływ wyboru algorytmu VAD na skuteczność rozpoznawania

Celem testu jest porównanie skuteczności zastosowania dwóch typów algorytmów wyboru początku i końca słowa (VAD-1SVM i MRSED) przy różnym stosunku sygnału do szumu.

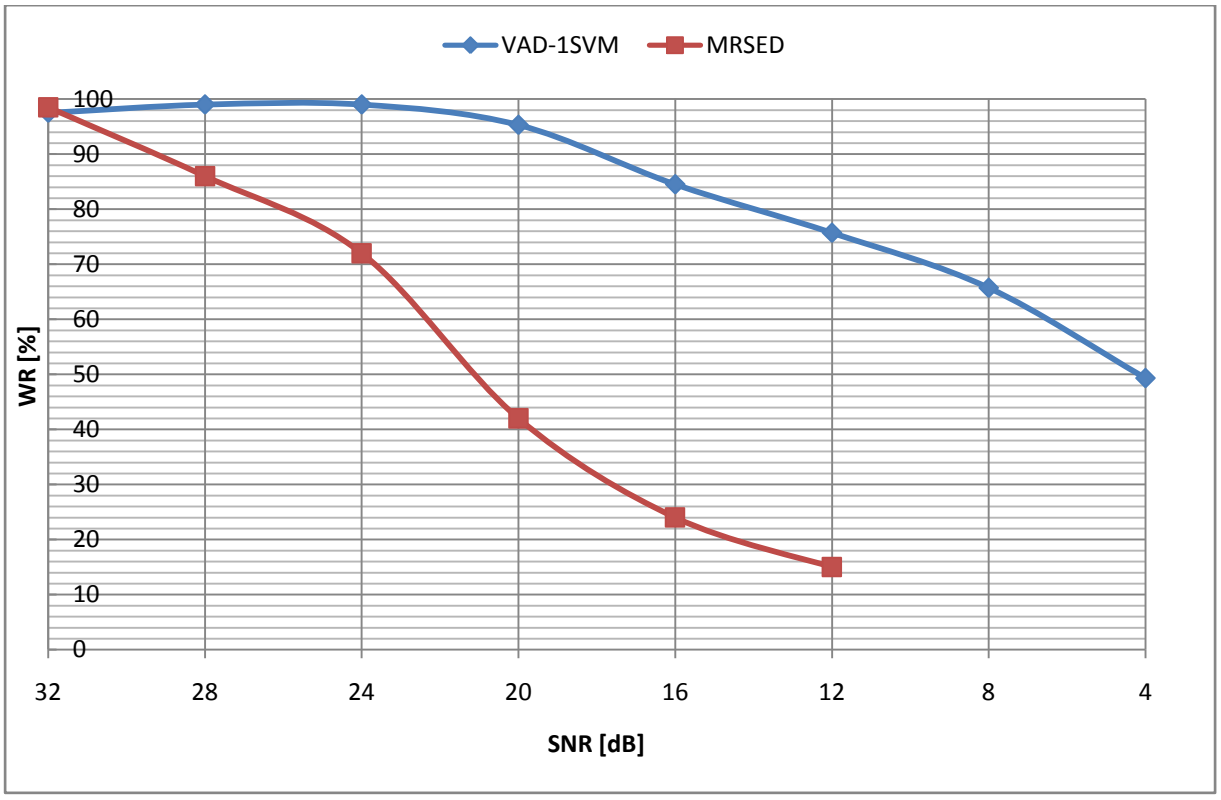
Warunki testu:

Baza danych	WBI
Liczba klas	20
Uczenie [liczba plików]	200
Testy [liczba plików]	200
Rozdz. pomiaru	0,5%

Wyniki testu (oprócz wyników uzyskanych przy SNR = 28 i 24dB) dla algorytmu VAD-1SVM stanowią uśrednione wyniki testów skuteczności rozpoznawania, które zostały przepisane z Tab. 4.7 (kolumna BFCCsr). Pomiary przy SNR = 28 i 24dB wykonano jednokrotnie.

Tab. 4.17 Wpływ wyboru algorytmu VAD na skuteczność rozpoznawania

VAD	VAD-1SVM (ν=0,15)	MRSED
log2(γ)	-13	-7
log2(C)	7	-1
CVGS [%]	97,5	98,5
WR(SN) [%]	97,5	98,5
WR(28dB) [%]	99,0	86,0
WR(24dB) [%]	99,0	72,0
WR(20dB) [%]	95,3	42,0
WR(16dB) [%]	84,5	24,0
WR(12dB) [%]	75,7	15,0
WR(8dB) [%]	65,7	-
WR(4dB) [%]	49,3	-



Rys. 4.27 Wykres porównujący skuteczność algorytmów VAD-1SVM i MRSED przy różnym SNR

Wnioski z eksperymentu:

Wyniki z Tab. 4.17 i Rys. 4.27 ilustrują, jak kluczowym zagadnieniem jest właściwy wybór algorytmu początku i końca w systemie rozpoznawania mowy. Oba algorytmy działają z podobną skutecznością (różnica 1% na korzyść MRSED) w przypadku, gdy szum jest naturalny (w przypadku bazy WBI jest on na poziomie ok. 32dB). Jednak kiedy tylko poziom szumu zaczyna wzrastać, widać jak różny wpływ ma to na oba algorytmy. W przypadku MRSED skuteczność rozpoznawania bardzo szybko maleje. Największy spadek skuteczności rozpoznawania wystąpił pomiędzy SNR równym 24dB a 20dB i wynosił aż 30%. Zakładając, że akceptowalny poziom skuteczności rozpoznawania wynosi powyżej 80% okazuje się, że algorytm MRSED może pracować w środowisku, gdzie SNR wynosi powyżej 26dB. W przypadku algorytmu VAD-1SVM taką skuteczność system może osiągnąć pracując przy SNR = 14dB. Można ponadto zauważyć, że algorytm MRSED „nie działał" [15] przy $SNR < 12dB$.

Mimo że algorytm VAD-1SVM jest bardziej czasochłonny oraz potrzebuje większych nakładów obliczeniowych widać, że w przypadku pracy systemu przy większym poziomie szumu, okazuje się on bez porównania lepszym rozwiązaniem niż MRSED. Z drugiej strony, jeśli wiemy, że system będzie działał w środowisku o niskim zaszumieniu o SNR na poziomie ok. 30dB lub wyższym, to algorytm MRSED może być ciekawą alternatywą dla algorytmu VAD ze względu na jego prostotę i szybkość działania.

[15] Zapis „nie działał" oznacza, że algorytm MRSED nie mógł znaleźć początku lub końca wypowiedzi, dla co najmniej jednego z badanych w teście sygnałów.

4.6.14 Badanie skuteczności systemu dla wielu mówców

Celem testów jest zbadanie skuteczności systemu przy założeniu, że z systemu będzie korzystać wielu mówców zarówno na etapie uczenia, jak i późniejszej pracy systemu. Dodatkowo zostanie wykonany test porównawczy ukazujący znaczenie algorytmu VAD w procesie rozpoznawania.

Doboru głosów mówców do testów A, B i D dokonano zgodnie z opisem w podrozdz. 4.3.3, który jest zgodny z kluczem zaproponowanym w [106]. Wyjątkiem jest test C, gdzie zarówno na etapie uczenia jak i testowania użyto jedynie plików z głosami męskimi, aby móc sprawdzić, jak wpływa zawężenie grupy mówców na wyniki rozpoznawania.

Warunki testu:

Baza danych	CORPORA
Liczba klas	20

Do testów A, B i C, gdzie zastosowano algorytm VAD, dodano sztuczny szum na poziomie SNR = 45dB, natomiast w teście D, gdzie nie zastosowano algorytmu VAD, nie było takiej potrzeby.

W algorytmie VAD-1SVM początkową wartość współczynnika ν ustawiono na 0,15.

Tab. 4.18 Badanie skuteczności systemu dla wielu mówców

Oznaczenie testu	A	B	C	D
Głosy*	m, ż, d	m, ż, d	**m**	m, ż, d
VAD	VAD-1SVM	VAD-1SVM	VAD-1SVM	**same słowa bezVAD**
Uczenie [liczba plików]	420	420	**280**	420
Testy [liczba plików]	400	400	**260**	400
Rozdz. pomiaru	0,25%	0,25%	0,38%	0,25%
log2(γ)	-9	-9	-13	-13
log2(C)	5	5	7	7
CVGS [%]	90,2	-	90,0	94,3
Skalowanie	[-1,1]	**[0,1]**	[-1,1]	[-1,1]
WR(SN) [%]	-	-	-	93,8
WR(45dB) [%]	89,3	93,8	93,8	-
WR(36dB) [%]	87,5	91,3	91,9	93,5
WR(32dB) [%]	87,5	90,3	90,4	93,5
WR(28dB) [%]	85,3	87,0	85,0	93,5
WR(24dB) [%]	77,8	81,3	80,4	93,0
WR(20dB) [%]	74,5	78,3	73,8	92,8
WR(16dB) [%]	62,3	64,5	61,9	89,8
WR(12dB) [%]	47,8	52,0	48,5	86,5
WR(8dB) [%]	37,3	38,3	36,5	81,0
WR(4dB) [%]	27,3	26,8	25,4	69,5

* Głosy: m - męskie, ż - żeńskie, d - dziecięce

Wnioski z eksperymentu:

W Tab. 4.18 zdecydowanie najlepsze wyniki uzyskano w teście D, kiedy nie trzeba było wykonywać algorytmu wyboru początku i końca. Różnica w wynikach była tym większa, im większy był poziom zaszumienia osiągając przy SNR = 4dB o ponad 40% wyższą skuteczność w porównaniu z pozostałymi testami A, B i C.

Porównując testy A i B widać, że w przypadku testu B uzyskano nieco lepsze wyniki. Oba testy różnią się warunkami skalowania. Mimo, że metodę *cross-validation* w połączeniu z *grid-search*, mającą na celu optymalny wybór parametrów C i γ, wykonano wcześniej skalując dane w zakresie od -1 do 1, to w teście B użyto tych parametrów stosując skalowanie w zakresie od 0 do 1. Uzyskanie więc lepszych wyników w teście B może dziwić. Wydaje się, że przyczyną tego zjawiska może być to, że w tym przypadku mogło nastąpić nadmierne dopasowanie sieci do danych uczących. Zjawiska takiego nie zaobserwowano w przypadku testów z bazą WBI, gdzie jest tylko jeden mówca, lecz tam, gdzie uczenie i testy były wykonywane na wypowiedziach pochodzących od wielu osób.

Dość ciekawie przedstawiają się także wyniki testu C, który wykonano wyłącznie na głosach męskich. W porównaniu z testem A uzyskano lepsze wyniki przy SNR = 45, 36 32, 24 i 12 dB. Przy porównaniu jednak otrzymanych wyników z testem B widać, że lepsze wyniki test C uzyskał jedynie przy małym zaszumieniu: 45, 36 i 32 dB. Jeśli szum był większy (SNR od 28 do 4dB), lepsze wyniki uzyskano w teście B.

Podsumowując otrzymane wyniki wydaje się, że większa różnorodność głosów w przypadku, gdy jest wielu mówców nie ma tak dużego wpływu na wyniki rozpoznawania. Ważne jest jednak, aby zapewnić odpowiednią liczbę wektorów uczących reprezentujących te grupy mówców, które będą stosowane na etapie rozpoznawania.

4.6.15 Porównanie skuteczności systemu rozpoznawania wykorzystującego SVM z systemem opartym na ukrytych modelach Markowa

Celem testu jest porównanie skuteczności opracowanego systemu rozpoznawania izolowanych słów z użyciem sieci SVM z wynikami systemu wykorzystującego ukryte modele Markowa.

Podobnie, jak w poprzednim teście (podrozdział 4.6.14) doboru głosów mówców do badań dokonano zgodnie z kluczem zamieszczonym w artykule [106]. Różnicę stanowi to, że przedmiotem rozpoznawania są 33 litery alfabetu z bazy nagrań CORPORA*: „a”, „ą”, „be”, „ce”, „cie”, „de”, „e”, „ę”, „ef”, „ge”, „ha”, „i”, „jot”, „ka”, „el”, „eł”, „em”, „en”, „eń”, „o”, „pe”, „ku”, „er”, „es”, „eś”, „esz”, „te”, „u”, „wu”, „y”, „zet”, „ziet” i „żet”*. Zbiór uczący tworzą nagrania 21 osób, a zbiór testowy tworzą nagrania 20 osób zgodnie z opisem zawartym w podrozdziale 4.3.3.

Sam proces uczenia jak i testowania sieci SVM wykonano z użyciem skryptu easy.py z [84]. Ze względu na to, że w artykule prof. Grocholewskiego [106] nie zastosowano w testach algorytmu VAD, pominięto tą część systemu także w tym teście, wykonując eksperyment na tych samych danych uczących i testowych co w artykule.

Warunki testu:

Baza danych	CORPORA
Etykiety	**Litery alfabetu**
Liczba klas	33
Uczenie [liczba plików]	693
Testy [liczba plików]	660
Uczenie [liczba mówców]	21
Testy [liczba mówców]	20
Rozdz. pomiaru	0,15%
Algorytm VAD	Brak

Doboru plików do testu dokonano zgodnie z [106]**.**

Przeprowadzono 3 testy proponowanego systemu. W teście A zastosowano częstotliwość próbkowania wynoszącą 8kHz, 5-stopniową metodą *cross-validation* w połączeniu z *grid-search*, 10 współczynników cepstralnych, a jako metodę parametryzacji wybrano BFCC. Test B różnił się tym od pierwszego, że zwiększono częstotliwość próbkowania do 16kHz, zastosowano dzielenie wektorów na 9, a nie jak dotychczas 5 części oraz zwiększono liczbę współczynników cepstralnych o 2 do 12. W teście C w stosunku do B jedyną modyfikacją była zmiana metody parametryzacji na MFCC.

Tab. 4.19 Testy skuteczności rozpoznawania liter alfabetu przez system rozpoznawania mowy z użyciem sieci SVM

Oznaczenie testu	A	B	C
Fp [kHz]	8	**16**	16
Met. parametryzacji	BFCC	**BFCC**	**MFCC**
M	240	300	300
N	80	100	100
p	10	12	12
LDCV	5	9	9
log2(γ)	-9	-11	-15
log2(C)	7	5	11
CVGS [%]	76,9	79,2	76,0
WR [%]	**84,4**	**86,2**	**84,8**

Badanie ma na celu porównanie nie tylko obu metod parametryzacji, lecz jest próbą porównania skuteczności dwóch różnych systemów rozpoznawania mowy. W [106] zastosowano np. inną metodę parametryzacji niż w pracy. Ogólny wstępny układ przetwarzania mowy zastosowany w teście prof. Grocholewskiego opisanym w [106] wygląda następująco:

- częstotliwość próbkowania - 16kHz
- preemfaza sygnału ze współczynnikiem 0,97,
- obliczanie co 10ms transformaty Fouriera z oknem Hamminga o długości 25ms,
- przekształcenie każdego z widm chwilowych do postaci 24 zachodzących na siebie filtrów o kształcie trójkątnym równo rozłożonych na skali melowej,
- przekształcenie amplitud sygnałów na wyjściach filtrów ze skali liniowej do logarytmicznej,

- obliczenie 12 współczynników cepstralnych za pomocą dyskretnej transformaty cosinusowej przeprowadzonej na sygnałach wyjściowych z filtrów oraz energii sygnału,
- obliczenie 13 pierwszych pochodnych i 13 drugich pochodnych dla współczynników cepstralnych oraz energii opisujących zmienność sygnału,
- zastosowanie liftru sinusoidalnego.

Ostatecznie sygnał mowy został opisany w przestrzeni 39-wymiarowej.

System bazowy rozpoznawania wykorzystuje modele 33 fonemów występujących w literach alfabetu. Punktem wyjścia były niezależne od kontekstu niejawne modele Markowa o najprostszej trzystanowej topologii typu "z lewa na prawo" i o połączeniach między sąsiednimi stanami (bez "przeskoków").

Tab. 4.20 Wyniki skuteczności rozpoznawania liter alfabetu przez system rozpoznawania mowy z użyciem HMM (dane z [106])

Metoda	System bazowy	Mod 1 (3 mixtures)	Mod 2 (CMN)	Mod 3 (trifony)	System "słuchowy"
WR [%]	83,3	86,4	74,4	89,1	98

W Tab. 4.20 znajdują się wyniki testów zamieszczone na podstawie [106], a zawarte skróty oznaczają odpowiednio:

- mod 1 - *3 mixtures* - modyfikacja systemu bazowego polegająca na zwiększeniu liczby rozkładów normalnych dla każdego ze stanów z jednego do trzech,

- mod 2 - CMN *Cepstral Mean Normalization*) – modyfikacja systemu bazowego polegająca na zastosowaniu odejmowania od wektorów współczynników cepstralnych wektora otrzymanego przez uśrednienie wszystkich wektorów wypowiedzi,

- mod 3 - *trifony* – modyfikacja w stosunku do bazowego systemu polegająca na zastąpieniu modeli fonemów niezależnych od kontekstu modelami fonemów zależnych od kontekstu,

- system „*słuchowy*" oznacza skuteczność rozpoznawania ludzkiego systemu rozpoznawania mowy, który był badany na grupie studentów.

Szczegółowy opis testów, których wyniki przedstawiono w Tab. 4.20 można znaleźć w [106].

Wnioski z eksperymentu:

Wyniki rozpoznawania zawarte w Tab. 4.19 i Tab. 4.20 są na zbliżonym poziomie osiemdziesięciu kilku procent. Jedynie wynik dla systemu HMM (3 mixtures) znacznie odbiega od średniej osiągając skuteczność poniżej 75%. Bezkonkurencyjne wyniki uzyskano z naturalnym systemem słuchowym człowieka, gdzie poziom błędu wynosił jedynie 2%. Jak widać nie ma w chwili obecnej systemu rozpoznawania mowy, który rozpoznawałby mowę lepiej niż człowiek i pewnie jeszcze przez jakiś czas taki stan rzeczy się utrzyma.

Porównując wyniki z Tab. 4.19 i Tab. 4.20 widać, że proponowany w pracy system SVM uzyskał w każdym z badanych przypadków wyższą skuteczność niż system bazowy HMM. Jednakże wprowadzenie modyfikacji do systemu bazowego spowodowało, że skuteczność zmodyfikowanego systemu uzyskała nieco lepsze wyniki w porównaniu do testu B z Tab. 4.19 odpowiednio o 0,2% (dla Mod 1 - 3 mixtures) i 2,9% (dla Mod 3 - trifony).

Przy analizie wyników zawartych w Tab. 4.19 widać, że podwojenie częstotliwości próbkowania, zwiększenie liczby do 9 w technice *cross-validation* oraz zwiększenie liczby

współczynników cepstralnych do 12, zwiększyło skuteczność rozpoznawania o 2,2%. Z kolei zmiana metody parametryzacji z BFCC (test B) na MFCC (test C) nie przyniosła poprawy, lecz pogorszenie wyników o 1,4%.

Porównując oba systemy należy zwrócić uwagę, że wyniki testu A w Tab. 4.19 z użyciem sieci SVM otrzymano dla systemu, gdzie $Fp = 8kHz$, a liczba parametrów wynosiła jedynie 10. Mimo że wynik był nieznacznie niższy od najlepszej z badanych metod HMM przedstawionych w Tab. 4.20, to należy mieć na uwadze, że był on osiągnięty przy $Fp = 16kHz$ oraz z użyciem 39 parametrów.

4.7 Podsumowanie eksperymentów i ostateczny opis parametrów systemu rozpoznawania mowy z użyciem sieci SVM

Na podstawie testu A i B z Tab. 4.19 zdecydowano, że standardową częstotliwością próbkowania stosowaną przy rozpoznawaniu mowy będzie 8kHz. Dwukrotne zwiększenie częstotliwości, powodujące znaczne zwiększenie liczby obliczeń i niezbędnej pamięci, sprawia, że staje się nieopłacalne stosowanie częstotliwości 16kHz lub wyższej.

Najlepsze z badanych metod parametryzacji okazały się metody oparte o parametry cepstralne: BFCC, MFCC i LPCC. Ze względu na to, że wyniki tych metod okazały się zbliżone, autor nie chce w sposób jednoznaczny oceniać, która z tych metod jest najlepsza. Jednocześnie dokonując wyboru jednej z metod do projektowanego systemu, proponuje wykorzystać parametry BFCC (ang. *Bark Frequency Cepstrum Coefficients*) jako metodę parametryzacji na etapie rozpoznawania.

Przeprowadzone badania wskazują, że najkorzystniej pod względem czasu obliczeń i wpływu na wyniki skuteczności rozpoznawania systemu jest zastosować uzupełnianie wektorów parametrów zerami do najdłuższego z wektorów, dlatego taką też metodę wybrano do projektu systemu, rezygnując z metody skracania do najkrótszego z wektorów i normalizacji względem stałej liczby próbek.

Kluczem do stworzenia skutecznego systemu rozpoznawania mowy z użyciem sieci SVM na etapie rozpoznawania jest właściwy dobór parametru jądra γ oraz stałej C. Sama zmiana jednego z parametrów bez uwzględnienia wartości drugiego nie jest skuteczna. Dlatego autor proponuje stosowanie metody *grid-search* na etapie uczenia sieci w celu znalezienia optymalnej pary parametrów C i γ. Najlepsze efekty metoda ta przynosi, jeśli jest poprzedzona metodą *cross-validation*, mającą za zadanie podział wektorów wejściowych na stałą liczbę podwektorów w celu sztucznego zwiększenia liczby wektorów uczących. Ze względu na to, że większa liczba podwektorów wydłuża dodatkowo czas uczenia, autor proponuje zastosowanie dzielenia na 5 podwektorów.

Autor dopuszcza także możliwość, że uczenie sieci będzie się odbywać z wartościami domyślnymi, tzn. $\gamma = \frac{1}{k}$, gdzie k oznacza długość wektora uczącego, a C będzie wynosić 1 (lub więcej). Zostaje wtedy zlikwidowany najdłuższy etap uczenia sieci, czyli doboru optymalnej pary parametrów C i γ. Może to jednak powodować, że sieć nie będzie działać w sposób optymalny. Wydaje się, że salomonowym rozwiązaniem jest dodanie procesu CVGS jako opcji, która będzie włączana np. przez użytkownika systemu lub automatycznie, dopiero gdy otrzymywane wyniki skuteczności systemu nie będą zadowalające.

Mimo że jadro sigmoidalne, liniowe i radialne uzyskały podobny poziom rozpoznawania to właśnie to ostatnie uzyskało najlepsze wyniki w testach w środowisku o największym badanym zaszumieniu SNR = 4dB. Było to podstawą, aby wybrać właśnie jądro radialne do tworzonego systemu rozpoznawania mowy.

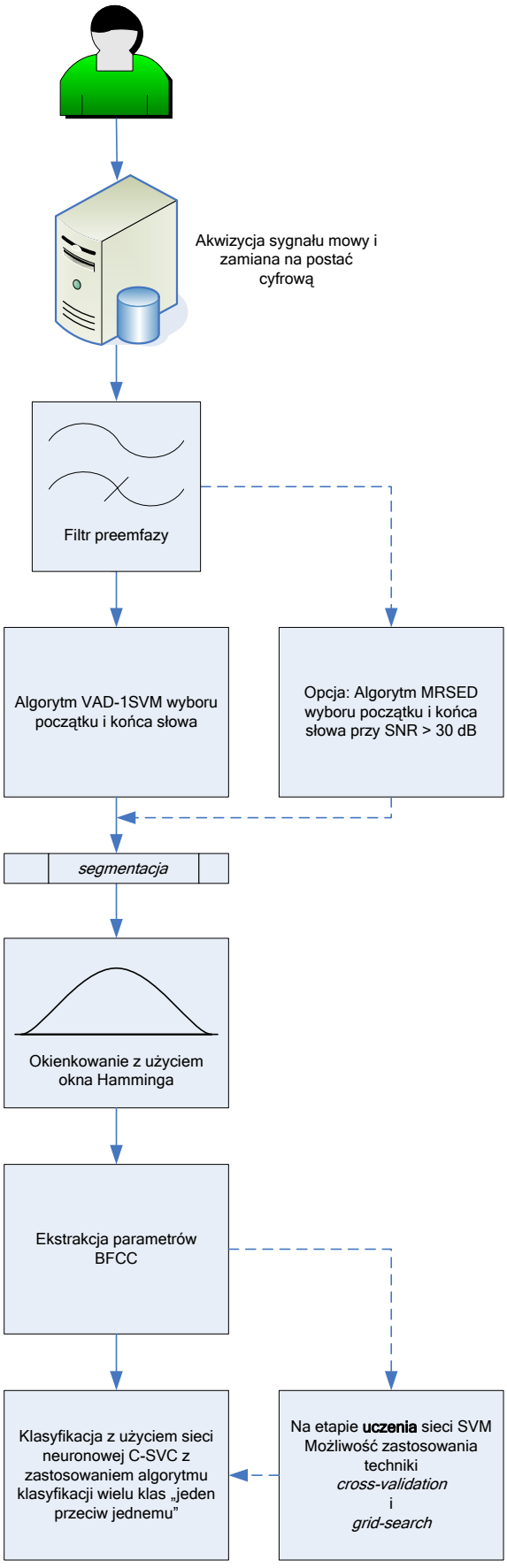
Oczywiste jest, że im większy jest słownik, tym trudniej dokonywać właściwej klasyfikacji słów poddawanych rozpoznawaniu. Przeprowadzone testy wskazały, że przy liczbie klas wynoszącej 20 skuteczność rozpoznawania nie spadła znacznie w porównaniu z mniejszą liczbą klas. Mimo że większość testów została wykonana dla tej liczby klas, to nic nie stoi na przeszkodzie, aby testować system także z większą liczbą klas. Przykładem może być tu podrozdział 4.6.15, w którym klasyfikacja odbywała się z użyciem 33 liter alfabetu.

Liczba powtórzeń danej klasy na etapie uczenia jest także jednym z kluczowych parametrów mających bezpośredni wpływ na wyniki rozpoznawania. Zbyt mała liczba powtórzeń, np. 5, może prowadzić do zaniżonych wyników. Należy zdawać sobie jednak sprawę, że wraz ze wzrostem liczby powtórzeń, zwiększa się także liczba wektorów podtrzymujących, które tworzą model sieci SVM. Autor proponuje więc stosowanie 10 powtórzeń na etapie uczenia każdej z klas. Jeśli z jakiegoś powodu byłoby to niemożliwe, zaleca się wtedy zastosowanie metody *cross-validation* z podziałem na większą liczbę podklas, w celu sztucznego zwiększania liczby wektorów uczących. Może być to przydatne np. w sytuacji, gdy nowy użytkownik chce szybko zacząć korzystać z systemu i nie chce lub nie ma czasu na wielokrotne powtarzanie tego samego słowa.

Jak pokazano w Tab. 4.17 i Rys. 4.27 właściwy wybór początku i końca wypowiedzi jest kluczowym elementem procesu rozpoznawania mowy. W opinii autora jest to jeden z najważniejszych części automatycznego systemu mowy, bez którego właściwie niemożliwe jest skuteczne rozpoznawanie. W pracy porównywane były dwa własne algorytmy wyboru początku i końca: jeden prosty MRSED, bazujący na pomiarze energii sygnału oraz bardziej skomplikowany, VAD-1SVM. W opinii autora obie metody mają zastosowanie w systemie rozpoznawania izolowanych słów.

Mimo że oba algorytmy uzyskały podobną skuteczność, jeśli stosunek sygnału do szumu był wyższy niż 30dB, to autor proponuje wykorzystać algorytm MRSED ze względu na jego prostotę i szybkość działania. W przypadku jednak, gdy system ma pracować przy większym zaszumieniu, autor proponuje zastosować bardziej rozbudowany algorytm VAD-1SVM, który znacznie lepiej radzi sobie w środowisku zaszumionym.

Schemat blokowy działania całego zaprojektowanego systemu w dużym uproszczeniu przedstawiono na Rys. 4.28. Ma on jedynie charakter poglądowy i nie zawiera wielu bardziej szczegółowych procesów, aby nie zaciemniać schematu.



Rys. 4.28 Uproszczony schemat zaprojektowanego systemu rozpoznawania izolowanych słów z użyciem sieci neuronowych SVM

W Tab. 4.21 przedstawiono końcowe zestawienie parametrów tworzących system rozpoznawania mowy z użyciem sieci SVM.

Tab. 4.21 Ostateczne parametry systemu rozpoznawania mowy z użyciem sieci SVM

Etap	Nazwy i oznaczenia parametrów lub	Wartość parametrów / metody
Akwizycja sygnału	Częstotliwość próbkowania [kHz]	8
	Ilość bitów	16
	Ilość kanałów	1
Preemfaza	Transmitancja filtru preemfazy *H(z)*	$1-az^{-1}$
	Współczynnik *a* w filtrze preemfazy	0.95
Wybór początku i końca słowa	Typ algorytmu VAD[16]	MRSED dla SNR ≥ 30dB oraz VAD-1SVM dla SNR < 30dB
	Wartość początkowa współczynnika v dla algorytmu VAD-1SVM	0,15
Ramkowanie i okienkowanie	Długość ramki *N [w próbkach]*	240
	Skok ramki *M [w próbkach]*	80
	Typ okna	Hamminga
Parametryzacja	Rząd predykcji p.	10.
	Metoda obliczania współczynników filtru predykcyjnego	Metoda autokorelacyjna z użyciem algorytmu Levinsona-Durbina
	Metoda parametryzacji	BFCC (ang. *Bark Frequency Cepstrum Coefficients*)
	Częstotliwości pasma filtrów [Hz]	Od 0 do 4000
	Liczba pasm spektralnych (ang. *warped spectra bands*)	40
	Współczynnik eksponenty w lifteringu	0.6
Wektory parametrów	Długość wektorów podawanych na wejście sieci SVM na etapie uczenia i testów	Uzupełnianie zerami do najdłuższego z wektorów
	Liczba powtórzeń danej klasy na etapie uczenia	10
Parametry sieci SVM	Typ sieci SVM	C-SVC (ang. *C-Support Vector Classification*)
	Typ jądra	Radialne
	Algorytm do klasyfikacji wielu klas	Jeden przeciw jednemu
	Skalowanie	Tak [-1,1]
	CVGS	Tak – technika *cross-validation* z podziałem na 5 podgrup wraz z metodą *grid-search*. Alternatywnie można metodę CVGS zastosować jako opcję i wprowadzić w momencie, gdy ustawienia domyślne nie przynoszą satysfakcjonujących wyników.
	C	Dobierane na podstawie CVGS. W przypadku braku CVGS proponuje się $C > 1$.
	γ	Dobierane na podstawie CVGS. Jeśli nie zastosowano CVGS proponuje się $\gamma = 1/k$.

[16] Szczegółowy opis parametrów algorytmu MRSED przedstawiono w podrozdz. 4.5.1 a algorytmu VAD-1SVM w podrozdz. 4.5.7.

Projektowany system może działać zarówno przy założeniu, że będzie korzystać z niego jeden mówca, jak i wtedy, gdy uczenie jak i późniejsza praca systemu będzie wykonywana z użyciem wielu mówców. Co ważne, system może pracować nawet wtedy, gdy mówcy są różnej płci i wieku. Ważne jest jednak, w przypadku pracy z większą liczbą użytkowników, aby zapewnić odpowiednią ilość danych uczących pochodzących od tych grup użytkowników, które reprezentują cechy przyszłych użytkowników systemu.

5. Zakończenie

5.1 Dalsze możliwości rozwinięcia systemu

Autor nie twierdzi, że jego system jest idealny, lub że charakteryzuje go najwyższa skuteczność. Ze względu na to, że ma on budowę modułową, możliwe jest ciągłe udoskonalanie poszczególnych jego elementów. Intencją autora było raczej przedstawienie, że możliwe jest stworzenie efektywnego systemu rozpoznawania izolowanych słów z wykorzystaniem sieci SVM aż w dwóch różnych częściach systemu, który będzie działał także w środowisku zaszumionym.

Dalsze prace rozwojowe powinny skupić się na takiej modyfikacji parametrów systemu, aby jego skuteczność była jeszcze wyższa przy jednoczesnej minimalizacji zasobów obliczeniowych i pamięciowych. Dotyczy to praktycznie każdego z elementów systemu. W przyszłości można by zbadać inne metody parametryzacji sygnału mowy takie jak np. RASTA-PLP. Ciągle także trwają prace nad opracowywaniem nowych metod klasyfikacji w sieciach SVM przy założeniu, że rozpoznawanych jest wiele klas. Z pewnością będą powstawać także nowe typy sieci SVM, być może o lepszej skuteczności niż prezentowane w rozprawie.

Ogólnie wiadomo, że problemem dla sieci neuronowych jest to, że wektory wejściowe muszą być równej długości, gdy w rzeczywistości tak nie jest. Stosuje się więc różnego typu metody zrównywania wektorów. Mimo że w pracy podejmowano pewne próby mające na celu normalizację długości sygnału, to nie osiągnięto satysfakcjonujących wyników, decydując się na uzupełnienia wektorów parametrów wejściowych zerami do najdłuższego z nich. Wydaje się, że jest to problem niezwykle istotny, któremu powinna być poświęcona szczególna uwaga. Pojawiają się pewne rozwiązania, które radzą sobie z nim na poziomie jądra sieci SVM, np. jądro DTAK (ang. *Dynamic Time Alignment Kernel*) [115], [116]. Autor uważa, że dalsze prace w tym kierunku mogłyby okazać się bardzo interesujące.

Szczególną unikalną cechą stworzonego systemu rozpoznawania mowy jest jego elastyczność. Możliwe jest zastosowanie dwóch typów algorytmów wyboru początku i końca słowa o różnej złożoności, w zależności od poziomu zaszumienia. Dodatkowo autor proponuje wprowadzenie metody *cross-validation* w połączeniu z *grid-search* tylko w momencie, gdy proponowane domyślne wartości parametrów sieci SVM nie przyniosą zakładanego poziomu skuteczności. Jest to zgodne z mądrym powiedzeniem, że „nie ma sensu wyciągać armaty na wróbla”.

Kolejnym krokiem powinno być jeszcze lepsze dopasowanie zaprojektowanego systemu do środowiska, czyli zwiększenie jego możliwości adaptacyjnych. Można by np. dokonać dodatkowych badań w celu znalezienia optymalnej wartości parametru v w algorytmie VAD-1SVM oraz skoku,

o który zwiększana jest wartość v w kolejnym kroku w zależności od zaszumienia. Wydaje się, że wartości te mogą być różne w zależności od stosunku sygnału do szumu.

Zaprojektowany system mógłby w przyszłości zostać zaadoptowany do takich celów jak np. głosowy wybór numerów telefonicznych w telefonii komórkowej stacjonarnej czy VOIP. Ponadto z jego pomocą możliwe jest opracowanie systemu sterowania z użyciem kilkunastu do kilkudziesięciu komend, którym można by zarządzać już nie tylko komputerem czy prostą maszyną, lecz nawet inteligentnym domem. Kilkoma prostymi komendami np. *„cieplej”*, *„zimniej”*, *„jasno”*, *„ciemno”*, *„zamknij”*, *„zabezpiecz”*, *„podlej”*, itd. można już sterować wieloma czynnościami. Jeśli tylko uzależni się wykonanie określonego działania od poprawnego rozpoznawania dwóch następujących po sobie klas, to zdolności takiego systemu rosną lawinowo. Można wtedy powiedzieć np. *„wyłącz” „światło”* lub też *„wyłącz” „woda”* lub *„wyłącz’ „ogrzewanie”*. Jeśli liczba następujących po sobie słów zostanie zwiększona do trzech, to można już wydać komendę np. *„wyłącz” „światło” „łazienka”*, lub *„włącz” „ogrzewanie” „sypialnia”*. Systemy tego typu będą nieocenioną pomocą w szczególności dla osób niepełnosprawnych ruchowo, które dzięki tego typu rozwiązaniom będą mogły w sposób bardziej skuteczny kierować swoim życiem.

5.2 Podsumowanie wyników rozprawy

Stworzono oryginalny, kompletny system rozpoznawania izolowanych słów z możliwością pracy w środowisku zaszumionym, który może być wykorzystywany zarówno przez jednego, jak i wielu mówców.

W czasie komputerowej symulacji pracy systemu, przy założeniu, że korzysta z niego jeden użytkownik oraz, że klasyfikacja będzie dokonywana spośród 20 różnych polskich imion, osiągnięto skuteczność (patrz Tab. 4.7 test BFCCsr) 97,5% przy naturalnym zaszumieniu (ok. 32dB), 95,3% dla SNR = 20dB i 84,5% przy 16dB. Co więcej, większy poziom szumu wcale nie oznacza, że rozpoznawanie nie jest możliwe. System testowany nawet przy SNR = 4dB nadal potrafi poprawnie rozpoznać średnio co drugie słowo. Należy podkreślić, że prezentowane wyniki dotyczą kompletnego systemu, w którym automatycznie dokonywany jest także wybór początku i końca słowa.

W przypadku pracy z wieloma mówcami otrzymano skuteczność prawie 90% przy SNR = 45dB oraz ponad 85% przy SNR = 28dB (patrz test A z Tab. 4.18).

Porównując skuteczność systemu (z wyłączeniem algorytmu wyboru początku i końca) z wynikami uzyskanymi przez prof. Grocholewskiego [106], wykorzystującego do testów system bazujący na ukrytych modelach Markowa, otrzymano zbliżone wyniki na poziomie osiemdziesięciu kilku procent. Porównując najlepsze wyniki w obu testach różnica między nimi wynosiła jedynie 2,9% na korzyść HMM.

W opinii autora wybór początku i końca słowa jest jednym z najważniejszych elementów systemu rozpoznawania mowy. Co więcej, wydaje się, że jest on często niedoceniany np. w publikacjach wyników różnych testów, które pomijają ten etap podając wyniki badań, w których wykorzystywano słowa z oznaczonym wcześniej początkiem i końcem.

Autor prezentuje dwa algorytmy wyboru początku i końca słowa. Pierwszy z nich - MRSED powstał jako modyfikacja algorytmu Rabinera-Sambura i stanowi jego uproszczenie z pewnymi

modyfikacjami [108], nadając się świetnie jako prosty algorytm w sytuacji środowiska o niskim zaszumieniu (SNR > 30dB).

Drugi algorytm jest oryginalnym zastosowaniem stosunkowo nowego typu sieci One Class SVM [92] do wyboru początku i końca słowa. Mimo że przykłady zastosowania innych typów sieci SVM istnieją, to w opinii autora zastosowanie sieci One Class SVM ma takie zalety, których nie posiadają inne typy sieci neuronowych. Algorytm, w odróżnieniu od innych tego typu działających w oparciu o sieci SVM, uczy się wyłącznie na rzeczywistych danych w trybie on-line. Nie ma potrzeby uczenia sieci na danych pochodzących z innych sygnałów, nie trzeba też uczyć sieci przy różnym SNR oraz na szumach pochodzących z różnych środowisk. Sieć dostosowuje się do takiego poziomu szumu i jego rodzaju, jaki jest obecny dla konkretnego sygnału mowy. Ponadto, algorytm VAD-1SVM uczy się rozpoznawać poprawnie tylko jedną klasę – szum. Dzięki temu liczba danych uczących jest minimalna, co ma decydujący wpływ na szybkość działania algorytmu. W opinii autora, to właśnie autorski algorytm VAD-1SVM stanowi największą wartość naukową pracy.

Kolejną, chyba najważniejszą częścią każdego systemu rozpoznawania mowy jest blok klasyfikacji i podejmowania decyzji. Autor wykorzystał także na tym etapie sieci SVM, korzystając w tym wypadku z sieci C-SVC (ang. *C – Support Vector Classification*). Mimo że tego typu systemy istnieją, to należało dobrać optymalne parametry i metody, przy których system będzie działał najlepiej, co też autor na podstawie wielu eksperymentów (patrz podrozdz. 4.6) uczynił.

Podsumowując autor uważa, że udało się stworzyć kompletny, oryginalny system rozpoznawania mowy, wykorzystujący zarówno na etapie wyboru początku i końca słowa oraz na etapie klasyfikacji dwa różne typy sieci SVM, który działa ze skutecznością powyżej 80% przy SNR ≥ 16dB dla jednego mówcy oraz przy SNR ≥ 28dB, jeśli z systemu korzysta wielu różnych mówców. Aby możliwe było osiągnięcie tego celu należało wcześniej m.in.:

- zbadać możliwości zastosowania techniki wektorów podtrzymujących w systemie rozpoznawania izolowanych słów,
- stworzyć własną bazę danych imion WBI do badań systemu ze względu na fakt, że baza CORPORA okazała się niewystarczająca, kiedy z systemu miał korzystać jeden użytkownik,
- dobrać wszystkie elementy systemu rozpoznawania mowy na podstawie przeprowadzonych eksperymentów,
- opracować nowy algorytm wyboru początku i końca z wykorzystaniem SVM ze względu na niewystarczającą skuteczność algorytmu MRSED,
- dobrać optymalne parametry i metody na etapie rozpoznawania z użyciem techniki wektorów podtrzymujących,
- przeprowadzić komputerową symulację działania systemu i zbadać jego odporność na szum.

Dodatek A – Problem poszukiwania ekstremów metodą mnożników Lagrange'a

Metoda mnożników Lagrange'a

Metoda mnożników Lagrange'a [13][17], nazwana od nazwiska słynnego matematyka Josepha Louisa Lagrange'a, stosowana jest do znajdowania lokalnych ekstremów funkcji wielu zmiennych w odniesieniu do jednego lub więcej ograniczeń (warunków). Metoda ta redukuje problem n zmiennych z k ograniczeniami do rozwiązywalnego problemu $n + k$ zmiennych bez ograniczeń. W metodzie tej wprowadza się nową, nieznaną zmienną, która jest skalarem, tzw. mnożnik Lagrange'a, dla każdego ograniczenia i formułuje się układ równań z użyciem mnożników Lagrange'a jako współczynnikami.

Niech funkcja $f \in \boldsymbol{R}^n$ oraz niech $g_k(\boldsymbol{x}) = 0$. Wtedy Lagrangian L zdefiniować można jako

$$L(\mathbf{x}, \boldsymbol{\alpha}) = f + \sum_{\mathrm{k}} \boldsymbol{\alpha}_{\mathrm{k}} \mathrm{g}_{\mathrm{k}}. \tag{A.1}$$

Kryteria optymalizacyjne oraz ograniczeń g_{k} są ściśle związane z ekstremami Lagrangianu:

$$\nabla_x L = 0 \Leftrightarrow \nabla_x f = -\sum_{\mathrm{k}} \boldsymbol{\alpha}_{\mathrm{k}} \nabla_x \mathrm{g}_{\mathrm{k}}, \tag{A.2}$$

oraz

$$\nabla_x L = 0 \Leftrightarrow \mathrm{g}_{\mathrm{k}} = 0. \tag{A.3}$$

Przykład:

Założono, że należy znaleźć maksymalne wartości dla

$$f(x, y) = x^2 y \tag{A.4}$$

z ograniczeniem, że współrzędne x i y leżą na kole o promieniu $\sqrt{3}$, czyli

$$x^2 + y^2 = 3. \tag{A.5}$$

Ze względu na fakt, że przyjęto tylko jedno ograniczenie, zostanie użyty tylko jeden mnożnik Lagrange'a α.

Aby zdefiniować funkcję $g(x, y)$ użyto ograniczenia

$$g(x, y) = x^2 + y^2 - 3. \tag{A.6}$$

Funkcja g jest równa zero wszędzie na okręgu o promieniu 3. Dlatego też dla tego przypadku można mnożyć $g(x, y)$ przez dowolny mnożnik, a następnie dodać do $f(x, y)$, nie zmieniając jej jednocześnie (dla tego przypadku).

Niech

$$\Phi(x, y, \alpha) = f(x, y) + \alpha g(x, y) = x^2 y + \alpha(x^2 + y^2 - 3). \tag{A.7}$$

[17] hasło: Lagrange multipliers

Krytyczne wartości Φ wystąpią, kiedy gradient Φ będzie wynosił zero. Pochodne cząstkowe w takim przypadku będą równe:

$$\frac{\partial \Phi}{\partial x} = 2xy + 2\lambda x = 0 \tag{A.8}$$

$$\frac{\partial \Phi}{\partial y} = x^2 + 2\lambda y = 0 \tag{A.9}$$

$$\frac{\partial \Phi}{\partial y} = x^2 + y^2 - 3 = 0\,. \tag{A.10}$$

Równanie (A.10) jest oryginalnym ograniczeniem natomiast równanie (A.8) prowadzi do $\lambda = -y$. Podstawiając otrzymany wynik do równania (A.9) otrzyma się

$$x^2 - 2y^2 = 0. \tag{A.11}$$

Podstawiając rozwiązanie równania (A.9) do równania (A.10) i rozwiązując je po y otrzyma się

$$y = \pm 1. \tag{A.12}$$

Rozwiązaniem układu równań są cztery punkty krytyczne na płaszczyźnie o współrzędnych:

$$\left(\sqrt{2}, 1\right); \left(-\sqrt{2}, 1\right);\ \left(\sqrt{2}, -1\right);\ \left(-\sqrt{2}, -1\right).$$

Podstawiając otrzymane współrzędne do funkcji Φotrzyma się następujące wyniki:

$$\Phi\left(\sqrt{2}, 1\right) = 2;\ \Phi\left(-\sqrt{2}, 1\right) = 2;\ \Phi\left(\sqrt{2}, -1\right) = -2;\ \Phi\left(-\sqrt{2}, -1\right) = -2,$$

stąd też funkcja osiąga minimum w punktach o współrzędnych $(\sqrt{2}, 1)$ oraz $(-\sqrt{2}, 1)$, a maksimum w dwóch pozostałych punktach.

Warunki Karush-Kuhn-Tucker'a

Uogólnieniem metody mnożników Lagrange'a są warunki Karush-Kuhn-Tucker'a, znane także jako warunki KKT. Określają one niezbędne warunki do znalezienia optymalnego rozwiązania w technice nieliniowego programowania. Niezbędne warunki dla problemu nierówności z ograniczeniami po raz pierwszy zostały opublikowane w pracy magisterskiej przez W. Karush'a [117], a później na nowo opublikowane wraz Haroldem W. Kuhn'em i Albertem W. Tucker'em w [118].

Założono następujący problem nieliniowej optymalizacji:

$$\min\left[f(x)\right], \tag{A.13}$$

w odniesieniu do $g_i(x) \leq 0$, $h_j(x) = 0$, gdzie $f(x)$ jest funkcją, którą należy zminimalizować. Funkcje $g_i(x)$ (dla $i = 1,2,\dots,m$) są nierównościowymi ograniczeniami, a funkcje $h_j(x)$ (dla $j = 1,2,\dots,l$) są równościowymi ograniczeniami. Ponadto m jest liczbą nierównościowych ograniczeń, a l ograniczeń równościowych.

Warunki konieczne

Założono, że funkcja celu $f:\mathbb{R}^n \to \mathbb{R}$ i funkcje z ograniczeniami $g_i:\mathbb{R}^n \to \mathbb{R}$ i $h_j:\mathbb{R}^n \to \mathbb{R}$ są ciągle różniczkowalne w punkcie $x^* \in S$. Jeśli x^* jest minimum lokalnym, to istnieją stałe $\lambda \geq 0$, $\mu_i \geq 0$ dla $i = 1,2,\ldots,m$ oraz ν_j dla $j = 1,2,\ldots,l$ takie, że:

$$\lambda + \sum_{i=1}^{m} \mu_i + \sum_{j=1}^{l} |\nu_j| > 0, \tag{A.14}$$

$$\lambda \nabla f(x^*) + \sum_{i=1}^{m} \mu_i \nabla g_i(x^*) + \sum_{j=1}^{l} \nu_j \nabla h_j(x^*) = 0, \tag{A.15}$$

$$\mu_i g_i(x^*) = 0, \tag{A.16}$$

dla wszystkich $i = 1,2,\ldots,m$.

Warunki regularności

Może się zdarzyć, że w warunkach koniecznych λ może wynosić zero. Takie przypadki określane są mianem „nienormalnych" (ang. *abnormal*) lub „zdegenerowanych" (ang. *degenerate*). Warunek konieczny nie bierze pod uwagę właściwości funkcji, lecz jedynie geometrię ograniczeń.

Istnieją pewne warunki regularności, które zapewniają, że rozwiązanie nie jest „zdegenerowane", tzn. dla których $\lambda \neq 0$. Zostaną one pokrótce przedstawione.

- Warunek liniowej niezależności (ang. *Linear Independence Constraint Qualification – LICQ*): gradienty aktywnych ograniczeń nierównościowych i gradienty ograniczeń równościowych są liniowo niezależne w x^*.
- Warunek Mangasarian'a-Fromowitz'a (ang. Mangasarian-Fromowitz constraint qualification – MFCQ): gradienty aktywnych ograniczeń nierównościowych i gradienty ograniczeń równościowych są dodatnio, liniowo niezależne w x^*.
- Warunek stałego rzędu (ang. Constant Rank Constraint Qualification – CRCQ): dla każdej z podgrup gradientów aktywnych ograniczeń nierównościowych i gradientów ograniczeń równościowych rząd w sąsiedztwie x^* jest stały.
- Warunek stałej, dodatniej, liniowej zależności (ang. Constant Positive Linear Dependance Qualification – CPLD): dla każdej z podgrup gradientów aktywnych ograniczeń nierównościowych i gradientów ograniczeń równościowych, jeśli jest dodatnio, liniowo zależna w x^*, to jest również dodatnio, liniowo zależna w sąsiedztwie x^*. $\nu_1, \nu_2, \ldots, \nu_n$ jest dodatnio, liniowo zależny, jeśli istnieje $a_1 \geq 0, \ldots, a_n \geq 0$ nie dla wszystkich zer takich, że

$$a_1\nu_1 + a_2\nu_2 + \ldots + a_n\nu_n = 0. \tag{A.17}$$

Ostatni z warunków dotyczy problemu tylko z nierównościowymi ograniczeniami. W takim przypadku istnieje punkt x taki, że $g_i(x) < 0$ dla $i = 1,2,\ldots,m$.

W rzeczywistości z warunku LICQ wynika MFCQ a z niego CPLD, oraz z LICQ wynika CRCQ a z niego z kolei CPLD, mimo że MFCQ nie jest równoważny CRCQ. W praktyce preferowane są słabsze warunki ograniczeń, ponieważ zapewniają silniejsze warunki optymalizacyjne.

Warunki wystarczające

Założono, że funkcja celu $f:\mathbb{R}^n \to \mathbb{R}$ i funkcje z ograniczeniami $g_i:\mathbb{R}^n \to \mathbb{R}$ są funkcjami wypukłymi, a funkcje $h_j:\mathbb{R}^n \to \mathbb{R}$ są transformowalne oraz niech istnieje minimum lokalne w punkcie x^*. Jeśli istnieją stałe $\mu_i \geq 0$ dla $i = 1,2,\ldots,m$ oraz ν_j dla $j = 1,2,\ldots,l$ takie, że:

$$\nabla f(x^*) + \sum_{i=1}^{m} \mu_i \nabla g_i(x^*) + \sum_{j=1}^{l} \nu_j \nabla h_j(x^*) = 0 \qquad \text{(A.18)}$$

oraz

$$\mu_i g_i(x^*) = 0 \text{ dla wszystkich } i = 1,2,\ldots,m, \qquad \text{(A.19)}$$

to punkt x^* jest minimum globalnym.

Dodatek B – Zawartość płyty CD-ROM

Na dysku CD-ROM utworzono 6 katalogów głównych.

Katalog **Biblioteka_libsvm-2.84** zawiera bibliotekę libsvm w wersji 2.84. Jest ona najważniejszą częścią oprogramowania, które wykorzystywano w pracy wszędzie tam, gdzie korzystano z sieci SVM.

Katalog **Gnuplot** zawiera program do tworzenia wykresów. Jest on używany wraz z oprogramowaniem dostępnym w katalogu **Python** do uruchamiania skryptów z biblioteki libsvm, np. skryptu easy.py do obliczania optymalnej pary parametrów C i γ z użyciem technik cross-validation i grid-search.

W katalogu **Rastamat** zapisano pliki matlaba, z których niektóre wykorzystano w pracy na etapie parametryzacji.

W katalogu **Sygnaly_mowy** zawarto pliki wav. Składa się on z dwóch podkatalogów: Własnej Bazy Imion zawierającej 400 plików oraz z wybranych 5 plików z Pierwotnej Bazy Danych, które zostały użyte w pracy.

W katalogu **Wybrane_robocze_pliki_mat** zamieszczono wybrane pliki, z których autor najwięcej korzystał pod koniec pracy. Są to pliki, które okazały się bardzo pomocne przy badaniach projektowanego systemu. Należy podkreślić, że są to pliki robocze autora a nie gotowe programy.

Dodatkowo zamieszczono plik **Przyklad_uzycia.rtf**, w którym przedstawiono przykładową propozycję użycia zamieszczonego na płycie oprogramowania.

W pliku **praca_doktorska_walendowski.pdf** autor zamieścił treść niniejszego opracowania.

Bibliografia

[1]. **K. H. Davis, R. Iddulph and S. Balashek**, Automatic Recognition of Spoken Digits, *J. Acoust. Soc. Am.*, 1952, Vol. 24, 6, pp. 637-642.

[2]. **H. F. Olson and H. Belar**, Phonetic Typewriter, *J. Acoust. Soc. Am.*, 1956, Vol. 28, 6, pp. 1072-1081.

[3]. **J. W. Forgie and C. D. Forgie**, Results Obtained from a Vowel Recognition Computer Program, *J. Acoust. Soc. Am.*, 1959, Vol. 31, 11, pp. 1480-1489.

[4]. **J. Suzuki and K. Nakata**, Recognition of Japanese Vowels - Preliminary to the Recognition of Speech, *J. Radio. Res. Lab.*, 1961, Vol. 37, 8, pp. 193-212.

[5]. **T. K. Vintsyuk**, Speech Descrimination by Dynamic Programming, *Kibernetika*, January-February 1968, Vol. 4, 2, pp. 81-88.

[6]. **F. Itakura**, Minimum Prediction Residual Applied to Speech Recognition, *IEEE Transactions Acoustics, Speech, Signal Proc.*, February 1975, Vols. ASSP-23, 1, pp. 67-72.

[7]. **L. R. Rabiner, et al.**, Speaker Independent Recognition of Isolated Words Using Clustering Techniques, *IEEE Transactions Acoustics, Speech, Signal Proc.*, August 1979, Vols. ASSP-27, pp. 336-349.

[8]. **L. R. Rabiner**, A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, *IEEE proceedings*, February 1989, Vol. 77, 2, pp. 257 – 287.

[9]. **R. P. Lippmann**, An introduction to Computing eighth Neural Nets, *IEEE ASSP Mag.*, April 1987, Vol. 4, 2, pp. 4-22.

[10]. **A. Weibel, et al.**, Phoneme Recognition Using Time-Delay Neural Networks, *IEEE Transactions Acoustics, Speech, Signal Proc.*, 1989, Vol. 37, pp. 393-404.

[11]. **C. H. Lee, et al.**, Acoustic Modeling for Large Vocabulary Speech Recognition, *Computer Speech and Language*, 1990, Vol. 4, pp. 127-165.

[12]. **R. Cole, et al.**, The Challenge of Spoken Language Systems: Research Directions for the Nineties, *IEEE Transactions on Speech and Audio Processing*, January 1995, Vol. 3, 1, pp. 1-21.

[13]. Wikipedia, the free encyclopedia, [Online] http://en.wikipedia.org/wiki/.

[14]. **L. Rabiner and B.-H. Juang**, *Fundamentals of speech recognition*, Englewood Cliffs, New Jersey : Prentice Hall PTR, 1993.

[15]. **J. R. Deller, et al.**, *Discrete Time Processing of Speech Signals*, New Jersey : Prentice Hall, 1993.

[16]. **L. R. Rabiner and Sambur M. R.**, An algorithm for determining the endpoint of isolated utterances, *The Bell System Technical Journal*, February 1975, Vol. 54, 2, pp. 297 – 315.

[17]. **R. Tadeusiewicz**, *Sygnał mowy*, Warszawa : WKiŁ, 1988.

[18]. **Cz. Basztura**, *Rozmawiać z komputerem*, Wyd. Prac Naukowych „Format", Wrocław : 1992.

[19]. **A. V. Oppenheim**, *Sygnały Cyfrowe - Przetwarzanie i zastosowania*, Warszawa : WNT, 1982, strony 110-151.

[20]. **B. P. Bogert, et al.**, The quefrency analysis of time series for echoes: cepstrum, pseudo-autocovariance, cross-cepstrum, and saphe cracking, *Proceedings of the Symposium on Time Series Analysis*, Chapter 15, 20.

[21]. **S. B. Davis and P. Mermelstein**, Comparison of Parametric representations for Monosyllabic Word Recognition in Continuously Spoken Sentences, *IEEE Transactions Acoustics, Speech and Signal Processing*, 1980, Vols. ASSP-28, 4, pp. 375-366.

[22]. **H. Gish and M. Schmidt**, Text-independent speaker identification, *IEEE Signal Processing Mag.*, 1994, Vol. 11, pp. 18–32.

[23]. **L. R. Rabiner and B.-H. Juang**, An introduction to Hidden Markov Models, *IEEE ASSP Mag.*, January 1986, Vol. 3, 1, pp. 4–16.

[24]. **Koc Alpay**, *Acoustic Feature Analysis for Robust Speech Recognition*, M. Sc. Dissertation, Boğaziçi University, 2002.

[25]. **H. Hermansky**, Perceptual Linear Predictive (PLP) Analysis of Speech, *Journal Acoustical Society of America*, 1989, Vol. 87, 4, pp. 1738–1752.

[26]. **H. Hermansky and N. Morgan**, RASTA Processing of Speech, *IEEE Transactions on Speech and Audio Processing*, 1994, Vol. 2, 4, pp. 578-589.

[27]. **H. Hermansky, et al.**, Rasta-PLP speech analysis technique, *ICASSP*, 1992, pp. 121–124.

[28]. **H. Hermansky, et al.**, Compensation for the effect of the communication channel in auditory - like analysis of speech (RASTA-PLP), *in Proc. EUROSPEECH*, 1991, Vol. 3, pp. 1367-1370.

[29]. **Cz. Basztura**, *Źródła, sygnały i obrazy akustyczne*, Warszawa : Wkił, 1988, strony 218 –223.

[30]. **W. Borodziewicz i K. Jaszczak**, *Cyfrowe Przetwarzanie Sygnałów*, Warszawa : WNT, 1987, strony 123 - 142.

[31]. **P. Walendowski**, Use of non-stationary iterative methods to calculate LPC coefficients based on speech signal correlation, *International Conference on Signals and Electronic Systems. ICSES '06*, Conference proceedings, Łódź, September 17-20, 2006, Vol. 2, [Ed. by M. Tadeusiewicz, et al.], pp. 691-694

[32]. **M. R. Schroeder**, Direct (nonrecursive) Relations Between Cepstrum and Predictor Coefficients, *IEEE Transactions on acoustics, speech, and signal processing*, 1981, Vols. ASSP-29, 2, pp. 297-301.

[33]. **J. D. Markel and A. H. Gray**, *Linear Prediction of Speech*, New York : Springer-Verlag, 1976.

[34]. **M. Zbancioc and M. Costin**, *Using neural networks and LPCC to improve speech recognition*, *International Symposium on Signals, Circuits and Systems*, July 2003, Vol.2, pp. 445- 448.

[35]. **G. Kang and L. Fransen**, *Bit Rate Speech Encoder Based on Line-Spectrum-Frequency*, NRL Rep. 8857, Washington : National Research Laboratory, 1985.

[36]. **F. Itakura**, Line spectrum representation of linear predictive coefficients, *Transactions Committee Speech Research, Acoustical Soc. Japan*, 1975, Vol. S75, p. 34.

[37]. **J. P. Campbell**, Speaker Recognition: A Tutorial, *Proceedings of the IEEE*, September 1997, Vol. 85, 9, pp. 1437-1462.

[38]. **P. Walendowski**, Wpływ wyboru miar odległości i metod parametryzacji na skuteczność rozpoznawania sygnału mowy, *III Krajowa Konferencja Elektroniki, materiały konferencyjne*, Kołobrzeg, 16-18 czerwca 2004, Tom II, strony 593-598.

[39]. **L. R. Rabiner, A. Rosenberg and S. Levinson**, Considerations in dynamic time warping algorithms for discrete word recognition, *IEEE Transactions Acoustic Speech Signal Process*, 1978, Vol. 26, 6, pp. 575-582.

[40]. **C. S. Myers and L. R. Rabiner**, A comparative study of several dynamic time-warping algorithms for connected word recognition, *The Bell System Technical Journal*, September 1981, Vol. 60, 7, pp. 1389-1409.

[41]. **L. E. Baum and T. Petrie**, Statistical inference for probabilistic functions of finite state Markov chains, *Annals of Mathematical Statistics*, 1966, Vol. 37, pp. 1554-1563.

[42]. **J. K. Baker**, The DRAGON system - An overview, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, February 1975, Vol. 23, pp. 24-29.

[43]. **F. Jelinek, L. R. Bahl, and R. L. Mercer**, Design of a linguistic statistical decoder for the recognition of continuous speech, *IEEE Transactions on Information Theory*, May 1975, Vol. 21, pp. 250-256.

[44]. **F. Jelinek**, Continuos speech recognition by statistical methods, *Proceedings of the IEEE*, April 1976, Vol. 64, pp. 532-556.

[45]. **W. S. McCulloch and W. H. Pitts**, A logical calculus of ideas immanent in nervous activity, *Bull. Math. Biophysics*, 1943, Vol. 5, pp. 115-119.

[46]. **T. Kohonen**, *Self-Organization and Associative Memory*, Berlin : Springer-Verlag, 1987.

[47]. **T. Kohonen**, The Self-organizing Map, *Proceedings of the IEEE*, September 1990, Vol. 78, 9, pp. 1464-1480.

[48]. **T. Kohonen**, *Self-Organizing Feature Maps*, New York : Springer-Verlag, 1995.

[49]. **D. E. Rumelhart, G. E. Hinton and R. J. Williams**, Learning internal representations by error propagation, *Parallel distributed processing: Explorations in the Microstructures of Cognition*, 1986, Vol. 1.

[50]. **E. McDermott and S. Katagiri**, LVQ-based shift-tolerant phoneme recognition, *IEEE Transactions SP.*, 1991, Vol. 39, pp. 1398-1411.

[51]. **S. Katagiri**, Applications of Artificial Neural Networks (ANNs) to Speech Processing, [book auth.] J. Hwang and Y. Hu, *Handbook of NEURAL NETWORK SIGNAL PROCESSING*, CRC PRESS, 2002, 10.

[52]. **N. Morgan and H. Bourlard**, Continous speech recognition - an introduction to the hybrid HMM/connectionist approach, *IEEE Signal Processing Mag.*, 1995, Vol. 12, 3, pp. 25-42.

[53]. **W. Gerstner and W. Kistler**, *Spiking Neuron Models: Single Neurons, Populations, Plasticity, Cambridge*, University Press, 2002.

[54]. **G. Baudat and F. Anouar**, Generalized discriminant analysis using a kernel approach, *Neural Computation*, 2000, 12 (10), pp. 2385–2404.

[55]. **S. Mika, et al.**, Fisher discriminant analysis with kernels, *Neural Networks for Signal Processing IX, IEEE*, 1999, pp. 41–48.

[56]. ***S. Mika, et al.***, Invariant feature extraction and classification in kernel spaces, *Advances in Neural Information Processing Systems*, 2000, 12, pp. 526–532.

[57]. **V. Roth and V. Steinhage**, Nonlinear discriminant analysis using kernel functions, *Advances in Neural Information Processing Systems*, 2000, 12, pp. 568–574.

[58]. ***S. Mika, et al.***, Kernel PCA and de–noising in feature spaces, *Advances in Neural Information Processing Systems*, 1999, 11, pp. 536-542.

[59]. **B. Schölkopf, et al.**, Input space vs. feature space in kernel-based methods, *IEEE Transactions on Neural Networks*, 1999, 10(5), pp. 1000–1017.

[60]. **B. Schölkopf, A. J. Smola and K. R. Müller**, Nonlinear component analysis as a kernel Eigenvalue problem, *Neural Computation*, 1998, 10, pp. 1299–1319.

[61]. **C. Cortes and V. N. Vapnik**, Support vector networks, *Machine Learning*, 1995, 20, pp. 273–297.

[62]. **V. N. Vapnik**, *The Nature of Statistical Learning Theory*, New York : Springer-Verlag, 1995.

[63]. **V. N**. **Vapnik**, Statistical Learning Theory, New York : John Wiley & Sons, 1998.

[64]. **B. Schölkopf, et al.**, Comparison of viewbased object recognition algorithms using realistic 3D models, *Artificial Neural Networks*, 1996, pp. 251–256.

[65]. **C. J. C. Burges and B. Schölkopf**, Improving the accuracy and speed of support vector learning machines, *Advances in Neural Information Processing Systems*, 1997, pp. 375–381.

[66]. **H. Drucker, D. Wu and V. N. Vapnik**, Support vector machines for span categorization, IEEE *Transactions on Neural Networks*, 1999, Vol. 10, 5, pp. 1048–1054.

[67]. **T. Joachims**, Text categorization with support vectormachines: learning with many relevant features, *Proceedings of the European Conference on Machine Learning*, Berlin : Springer-Verlag, 1998, pp. 137–142.

[68]. **H.-C. Kim, et al.**, Constructing support vector machine ensemble, *Pattern Recognition*, 2003, 36, pp. 2757-2767.

[69]. **K. R. Müller, et al.**, Predicting time series with support vector machines, *Artificial Neural Networks — ICANN'97*, Berlin : Springer-Verlag, 1997, pp. 999–1004.

[70]. **A. Zien, et al.**, Engineering support vector machine kernels that recognize translation initiation sites, *Bioinformatics*, 2000, Vol. 16, 9, pp. 799–807.

[71]. **P. Janik, et al.**, Classification of Voltage Fluctuations Using SVM Network, *PSP Conference*, Slovenia, 2004, pp. 55-60.

[72]. **P. Janik**, *Identyfikacja zakłóceń jakości energii elektrycznej z zastosowaniem wybranych architektur sztucznych sieci neuronowych*, Rozprawa doktorska, Wrocław : Politechnika Wrocławska, 2005.

[73]. **S. Osowski**, Sieci neuronowe SVM w zastosowaniu do problemów regresji, *Przegląd Elektrotechniczny*, Październik 2002, strony 225-228.

[74]. **P. Clarkson and P. Moreno**, On the use of support vector machines for phonetic classification, *ICASSP99*, 1999, paper no. 2104.

[75]. **P. Niyogi, C. Burges and P. Ramesh**, Distinctive feature detection using support vector machines, *Proc. ICASSP99*, 1999, paper no. 1995.

[76]. **A. Ganapathiraju**, *Support vector machines for speech recognition*, PhD thesis, Mississippi : Mississippi State University, May 2002.

[77]. **R. Solera-Urena, et al.**, Robust ASR using Support Vector Machines, *Speech Communication*, 2007, 49.

[78]. **V. N. Vapnik**, *Estimation of Dependences Based on Empirical Data, Addendum 1*, New York : Springer-Verlag, 1982.

[79]. **W. B. Powell**, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, Wiley, 2007.

[80]. **V. N. Vapnik, S. Golovich and A. Smola**, Support vector method for function approximation, regression, estimation and signal processing, *Advances in neural information processing systems,* Cambridge, MA : MIT Press, 1997, pp. 281-287.

[81]. **S. Osowski**, Sieci neuronowe typu SVM w zastosowaniu do klasyfikacji wzorców, *Przegląd Elektrotechniczny*, Luty 2002, strony 29-36.

[82]. **K. S. Chua**, Efficient computations for large least square support vector machine classifiers, *Pattern Recognition Letters*, 2003, 24, pp. 75-80.

[83]. **I. N. Bronstein i K. A. Siemiendiajew**, *Matematyka – Poradnik Encyklopedyczny*, Warszawa : PWN, 2002.

[84]. **C.-C. Chang and C.-J. Lin**, *LIBSVM: a library for support vector machines*, [Online] September 2006, www.csie.ntu.edu.tw/~cjlin/libsvm.

[85]. **T. Cover**, Geometrical and statistical properties of system of linear inequalities with applications in pattern recognition, *IEEE Transaction on Electronic Computers*, Vol. 14, pp. 326-334.

[86]. **G. Guo, S. Z. Li and K. L. Chan**, Support Vector Machines for face recognition, *Image and Visio Computing*, 2001, 19, pp. 631-638.

[87]. **C.-W. Hsu and Lin, C.-J.**, A Comparison of Methods for Multiclass Support Vector Machines, *IEEE Transactions on Neural Networks*, March 2002, Vol. 13, 2, pp. 415-425.

[88]. **D. Anguita, A. Boni and S. Ridella**, A Digital Architecture for Support Vector Machines: Theory, Algorithm, and FPGA Implementation, *IEEE Transactions on Neural Networks*, September 2003, Vol. 14, 5, pp. 993-1009.

[89]. **C. Burges**, A tutorial on Support vector machines for pattern recognition, *Data Mining and Knowledge Discovery*, February 1998, 2, pp. 121-167.

[90]. **T. Górecki**, Metody konstrukcji jąder, *XXXIII Konferencja "Statystyka matematyczna"*, 3-7 Grudzień 2007, Prezentacja [Online] http://wisla2007.mat.umk.pl/referaty/Gorecki.pdf.

[91]. **C.-W. Hsu, C. C. Chang and C.-J. Lin**, *A Practical Guide to Support Vector Classification*, Available online http://wwwcsie.ntu.edu.tw/ćjlin/papers/guide/guide.pdf.

[92]. **B. Schölkopf, J. C. Platt and J. Shawe-Taylor**, Estimating the support of a high-dimensional distribution, *Neural Computation*, 2001, Vol. 13, 7, pp. 1443 - 1471.

[93]. **L. Bottou, et al.**, Comparison of classifier methods: a case study in handwriting digit recognition, *IEE Computer Society Press, International Conference on pattern recognition*, 1994, pp. 77-87.

[94]. **S. Kneer, L. Personnaz and G. Dreyfus**, Single layer learning revisited: a stepwise procedure for building and training a neural network, *Neurocomputing: Algorithms, Architectures and Applications*, 1990.

[95]. **J. Friedman**, *Another approach to polychotomous classification*, Department of Statistics, Stanford University, 1996, [Online at]: http://www-stat.stannford.edu/reports/friedman/poly.ps.z.

[96]. **U. Kreßel**, Pairwaise classification and support vector machines, *Advances in kernel Methods - Support Vector learning*, Cambridge, MA : MIT Press, 1999, pp. 255-268.

[97]. **J. C. Platt, et al.**, Large margin DAGs for multiclass classification, *Advances in Neural Information Processing Systems*, 2000, Vol. 12, pp. 547-553.

[98]. **C. Crammer and Y. Singer**, On the learnability and design of output codes for multiclass problems, *Computational Learning Theory*, 2000, pp. 35-46.

[99]. **I. V. Tetko, D. J. Livingstone and A. I. Luik**, Neural network studies, 1. Comparison of overfitting and overtraining, *J. Chem. Inf. Comput. Sci.*, 1995, Vol. 35, pp. 826-833, Available at http://www.vcclab.org/articles/jcics-overtraining.pdf.

[100]. MathWorks, Środowisko programistyczne Matlab, [Online] http://www.mathworks.com/.

[101]. Środowisko programistyczne Octave, [Online] http://www.gnu.org/software/octave/.

[102]. Oprogramownie do języka programowania Python, [Online] http://www.python.org/.

[103]. Oprogramownie do wykresów - Gnuplot, [Online] http://www.gnuplot.info/.

[104]. **S. Grocholewski**, *Baza nagrań sygnałów mowy polskiej CORPORA*, [płyta CD], Poznań : Instytut Informatyki, Politechnika Poznańska, 1997.

[105]. **P. Walendowski**, *Wybieranie numerów telefonicznych hasłem wypowiadanym przez człowieka*, Praca magisterska, Wrocław : Politechnika Wrocławska, 2002.

[106]. **S. Grocholewski**, The use of CORPORA for comparing ASR systems, *Speech and Language Technology*, 1999, Vol. 3, pp. 277-286.

[107]. Wikipedia, wolna encyklopedia, [Online] http://pl.wikipedia.org/wiki/.

[108]. **P. Walendowski**, The modification of Rabiner-Sambur Algorithm for Determining the endpoints of Isolated Words, *International Conference on Signals and Electronic Systems (ICSES)*, Conference proceedings, Poznań, 13-15 September 2004, [Ed. by M. Bartkowiak], pp. 401-404.

[109]. **A. Ganapathiraju, et al.**, Comparison of energy-based endpoint detectors for speech signal processing, Southeastcon '96, *'Bringing Together Education, Science and Technology', Proceedings of the IEEE*, April 11-14, 1996, pp. 500-503.

[110]. **A. Benyassine, et al.**, ITU-T Recommendation G.729 Annex B: A Silence Compression Scheme for Use with G.729 Optimized for V.70 Digital Simultaneous Voice and Data Applications, *IEEE Communications Magazine*, September 1997, pp. 64-73.

[111]. **D. Enqing, et al.**, Applying Support Vector Machines to Voice Activity Detection, *IEEE ICSP'02 Proceedings*, 2002, pp. 1124 - 1127.

[112]. **X. Xianbo and H. Guangshu**, An Incremental Support Vector Machine based Speech Activity Detection Algorithm, *Proceedings of the 2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*, September 2005, pp. 4224-4226.

[113]. **T. Cai and Y. Liang**, Robust endpoint detection based on one-class SVM, *Journal of Shenzhen Institute of Information Technology*, 2006, Vol. 6, 4, pp. 19 - 24.

[114]. **D. P. W. Ellis**, *PLP and RASTA (and MFCC, and inversion) in Matlab*, [Online] 2005, http://www.ee.columbia.edu/~dpwe/resources/matlab/rastamat/.

[115]. **H. Shimodaira, et al.**, Support vector machine with Dynamic Time-Alignment Kernel for speech recognition, *Proceeding Eurospeech*, 2001, pp. 1841-1844.

[116]. **H. Shimadaira, K. Noma and M. Nakai**, Dynamic time-alignment kernel in Support Vector Machine, *Advances in Neural Information Processing Systems 14*, 2002, Vol. 2, pp. 921-928.

[117]. **W. Karush**, Minima of Functions of Several Variables with Inequalities as Side Constraints, M. Sc. Dissertation, 1939.

[118]. **H. W. Kuhn and A. W. Tucker**, Nonlinear programming, *Proceedings of 2nd Berkley Symposium*, 1951, pp. 481-492.

[119]. **S. P. Lloyd**, Least Squares Quantization in PCM, *IEEE Transactions on information theory*, March 1980, Vols. IT-28, 2, pp. 129 – 137.

[120]. **Y. Linde, A. Buzo and R. M. Gray**, An algorithm for vector quantizer design, *IEEE Transactions on Communications*, January 1980, Vols. COM-28, pp. 84 – 95.

[121]. **A. Lapedes and R. Farber**, Nonlinear signal processing using neural networks: prediction and system modeling, Report, Los Alamos National Laboratory, 1987.

[122]. **B. Widrow**, 30 years of adaptative neural networks: perceptron, madaline and backpropagation, *Proceedings IEEE*, 1990, Vol. 78, pp. 1415-1442.

[123]. **A. J. Robinson**, An application of recurrent nets to phone probability estimation, *IEEE Transactions Neural Networks*, 1994, Vol. 5, 2, pp. 298-305.

[124]. **T. Kohonen, G. Barna and R. Chrisley**, Statistical pattern recognition with neural networks: Benchmarking studies, *Proc. IEEE Int. Conf. on Neural Networks, ICCN-88*, 1988, pp. I-61-I-68.

[125]. **T. Kohonen**, *Self-Organizing Maps*, Berlin : Springer-Verlag, 1997.

[126]. **R.-E. Fan, P.-H. Chen and C.-J. Lin**, Working set selection using second order information for training SVM, *Journal of Machine Learning Research 6*, 2005, pp. 1889-1918.

[127]. **B. Gas, J. L. Zarader and C. Chavy**, A new approach to speech coding: The neural predictive coding, *J. Adv. Comp. Intell.*, 2000, Vol. 4, 1, pp. 120-127.

[128]. **B. Gas, et al.**, Discriminant neural predictive coding applied to phoneme recognition, *Neurocomputing*, 2004, 56, pp. 141-166.

[129]. **A. Waibel, et al.**, Phoneme Recognition Using Time-Delay Neural Networks, *IEEE Transactions Acoustics, Speech, Signal Proc.*, 1989, Vol. 37, pp. 393-404.

Spis ilustracji

Rys. 2.1 Schemat blokowy systemu rozpoznawania mowy 12
Rys. 2.2 Schemat blokowy akwizycji sygnału mowy z wykorzystaniem karty dźwiękowej podłączonej do komputera 13
Rys. 2.3 Sygnał mowy podzielony na ramki o długości *N* i przesunięciu *M* 17
Rys. 2.4 Schemat blokowy wyznaczania cepstrum rzeczywistego sygnału 21
Rys. 2.5 Wykresy ilustrujące związek między skalą melową (rys. a) i barkową (rys. b) a klasyczną skalą częstotliwości 23
Rys. 2.6 Model generacji mowy 24
Rys. 2.7 Funkcja przekroju tuby akustycznej 31
Rys. 2.8 Zera wielomianów *P* i *Q* na płaszczyźnie *Z* 32
Rys. 2.9 Ilustracja działania algorytmu DTW 37
Rys. 3.1 Przykład problemu separacji w 2-wymiarowej przestrzeni dla dwóch klas. 41
Rys. 3.2 Struktura sieci neuronowej opartej na radialnych funkcjach bazowych 48
Rys. 3.3 Ilustracja problemu klasyfikacji w przestrzeni o trzech wymiarach 52
Rys. 3.4 Ilustracja działania metody *grid-search* 55
Rys. 4.1 Szum różowy w dziedzinie częstotliwości 62
Rys. 4.2 Algorytm wyznaczania początku słowa wykorzystujący pomiar energii sygnału 66
Rys. 4.3 Algorytm wyznaczania końca słowa wykorzystujący pomiar energii sygnału 67
Rys. 4.4 Wykres sygnału słowa *„dom”* z zaznaczonymi punktami początku i końca słowa w przypadku braku zewnętrznego szumu 68
Rys. 4.5 Przebieg czasowy (a) i energia w poszczególnych ramkach (b) słowa *„faraon”* z zaznaczonym początkiem i końcem wypowiedzi (brak zewnętrznego szumu, zastosowano filtr preemfazy) 69
Rys. 4.6 Wybór początku i końca sygnału dla wypowiedzi "*faraon*" przy różnym SNR (nie zastosowano filtru preemfazy) 69
Rys. 4.7 Wybór początku i końca sygnału dla wypowiedzi „*faraon*” przy różnym SNR 70
Rys. 4.8 Wybór początku i końca sygnału dla wypowiedzi „*Justyna Kowalska*” przy różnym SNR (nie zastosowano filtru preemfazy) 71
Rys. 4.9 Wybór początku i końca sygnału dla wypowiedzi „*Justyna Kowalska*” przy różnym SNR (po przejściu przez filtr preemfazy) 71
Rys. 4.10 Wybór początku i końca sygnału dla wypowiedzi „*szkoła*” przy różnym SNR 72
Rys. 4.11 Wybór początku i końca sygnału dla wypowiedzi „*szkoła*” przy różnym SNR 73
Rys. 4.12 Wykresy parametrów ΔLSF przed (a) i po (b) operacji podniesienia do kwadratu 75
Rys. 4.13 Wpływ filtru preemfazy na parametry ZCR sygnału niezaszumionego dla słowa „*faraon*”: a) znormalizowany sygnał mowy, b) parametry ΔZCR bez filtru preemfazy, c) parametry ΔZCR z filtrem preemfazy 76
Rys. 4.14 Wpływ filtru preemfazy na parametry ZCR przy SNR = 4dB dla słowa „*faraon*”: a) zaszumiony znormalizowany sygnał mowy, b) parametry ΔZCR bez filtru preemfazy, c) parametry ΔZCR z filtrem preemfazy 77
Rys. 4.15 Wpływ zastosowania operacji potęgowania na parametry ΔZCR dla słowa *„faraon”* w środowisku: a) braku zewnętrznego szumu, b) SNR = 20dB, c) SNR = 12dB oraz d) SNR = 4dB 78

Rys. 4.16 Przykład błędnego rozpoznania szumu jako sygnału wypowiedzi *„Ola Mucha*" w przypadku braku zewnętrznego szumu przy *v = 0,52* 83
Rys. 4.17 Ogólny schemat działania algorytmu VAD-1SVM 85
Rys. 4.18 Porównanie błędów wyboru początku wypowiedzi algorytmów MRSED i VAD-1SVM przy różnym SNR 90
Rys. 4.19 Porównanie błędów wyboru końca wypowiedzi algorytmów MRSED i VAD-1SVM przy różnym SNR 91
Rys. 4.20 Porównanie całkowitych błędów wyboru początku i końca wypowiedzi algorytmów MRSED i VAD-1SVM przy różnym SNR 91
Rys. 4.21 Wykres skuteczności rozpoznawania systemu dla różnych metod parametryzacji przy różnym SNR (jeden mówca, algorytm VAD) 101
Rys. 4.22 Różna długość tego samego słowa *„zero"* nagrana przez różne osoby przy *Fp = 8kHz* 103
Rys. 4.23 Wykres porównujący metodę uzupełniania wektorów zerami z metodą stałej liczby próbek przy różnym SNR 106
Rys. 4.24 Wykres wpływu pary parametrów *C* i *γ* dobieranych metodą *grid-search* na wyniki uczenia sieci SVM 110
Rys. 4.25 Wykres wpływu liczby klas na skuteczność rozpoznawania przy różnym SNR 112
Rys. 4.26 Wykres skuteczności rozpoznawania w zależności od liczby powtórzeń 114
Rys. 4.27 Wykres porównujący skuteczność algorytmów VAD-1SVM i MRSED przy różnym SNR 116
Rys. 4.28 Uproszczony schemat zaprojektowanego systemu rozpoznawania izolowanych słów z użyciem sieci neuronowych SVM 123

Spis tabel

Tab. 2.1 Typowe wartości parametrów używane do segmentacji sygnału 17
Tab. 2.2.2 Terminologia związana z widmem i cepstrum 21
Tab. 3.1 Tabela podobieństw i różnic pomiędzy tradycyjnymi sieciami neuronowymi a sieciami SVM 41
Tab. 3.2 Wybrane typy jąder 49
Tab. 4.1 Wyniki testów porównawczych algorytmu MRSED i VAD-1SVM 89
Tab. 4.2 Błąd średni skuteczności działania algorytmu MRSED przy różnym SNR 89
Tab. 4.3 Błąd średni skuteczności działania algorytmu VAD-1SVM przy różnym SNR 90
Tab. 4.4 Opis oznaczeń stosowanych przy opisie testów 94
Tab. 4.5 Porównanie wpływu parametrów cepstralnych: BFCC, MFCC i LPCC na wyniki rozpoznawania (wielu mówców, brak algorytmu VAD) 99
Tab. 4.6 Porównanie wpływu pochodnych parametrów LPC na wyniki rozpoznawania 99
Tab. 4.7 Porównanie wpływu parametrów cepstralnych: BFCC, MFCC i LPCC na wyniki rozpoznawania (jeden mówca, algorytm VAD) 101
Tab. 4.8 Porównanie wpływu pochodnych parametrów LPC na wyniki rozpoznawania (jeden mówca, algorytm VAD) 101
Tab. 4.9 Wpływ długości wektorów na wyniki rozpoznawania 104
Tab. 4.10 Porównanie metody uzupełniania 105
Tab. 4.11 Wpływ parametru C na skuteczność rozpoznawania i liczbę wektorów podtrzymujących 107
Tab. 4.12 Wpływ wsp. γ na skuteczność rozpoznawania i liczbę wektorów podtrzymujących 108
Tab. 4.13 Wyniki wpływy skalowania i CVGS na wyniki rozpoznawania 109
Tab. 4.14 Wpływ typu jądra na wyniki rozpoznawania 111
Tab. 4.15 Wpływ wielkości słownika na wyniki rozpoznawania 112
Tab. 4.16 Wpływ liczby powtórzeń na etapie uczenia na wyniki rozpoznawania 114
Tab. 4.17 Wpływ wyboru algorytmu VAD na skuteczność rozpoznawania 115
Tab. 4.18 Badanie skuteczności systemu dla wielu mówców 117
Tab. 4.19 Testy skuteczności rozpoznawania liter alfabetu przez 119
Tab. 4.20 Wyniki skuteczności rozpoznawania liter alfabetu przez system rozpoznawania mowy z użyciem HMM (dane z [106]) 120
Tab. 4.21 Ostateczne parametry systemu rozpoznawania mowy z użyciem sieci SVM 124