ROZPOZNAWANIE ZDAŃ W SYGNALE MOWY Z WYKORZYSTANIEM MODELU HMM

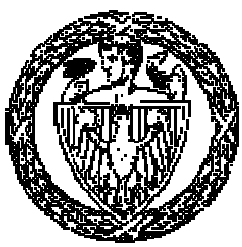
Paweł Przybysz
Włodzimierz Kasprzak

Raport IAiIS PW Nr 12-05

Warszawa, maj 2012 r.

POLITECHNIKA WARSZAWSKA
WYDZIAŁ ELEKTRONIKI I TECHNIK INFORMACYJNYCH
INSTYTUT AUTOMATYKI I INFORMATYKI STOSOWANEJ
ul. Nowowiejska 15 - 19, PL-00-665 WARSZAWA

Tel.: 0 22 234 7397, 0 22 825 52 80,
Fax: 0 22 825 37 19
E-mail: sekretariat@ia.pw.edu.pl



Streszczenie

Niniejszy raport jest drugim z serii raportów (piewrwszy to IAiIS PW 12-04 [KAS12]) dotyczącym badania przemysłowego n.t. klasyfikacji i rozpoznawania (izolowanych) zdań mówionych, wykonanego przez Autorów z Instytutu Automatyki i Informatyki Stosowanej Politechniki Warszawskiej. Raport przedstawia proces symbolicznego rozpoznawania słów kluczowych i zdań izolowanych w sygnale mowy w oparciu o model HMM (ukryty model Markowa).

Poprzedni raport [KAS12] opisuje niższe warstwy systemu analizy mowy: analizę akustyczną, kwantyzację wektorową cech ramek sygnału wspomaganą wiedzą fonetyczną i działanie prostego klasyfikatora DTW (tzw. klasyfikacja sekencji cech z „marszczeniem czasu"). Klasyfikator DTW jest znacznie jako mniej uniwersalnym rozwiązaniem od systemu rozpoznawania w modelem HMM, stanowiąc alternatywę jedynie w sytuacji ograniczonego słownika słów i małej liczby oczekiwanych zdań.

Przedmiot pracy jest już dobrze opracowany od strony teoretycznej. W literaturze przedmiotu znanych jest kilka typowych rozwiązań. Niniejszą pracę charakteryzuje autorskie ujęcie tematyki w postaci 3-poziomowego modelu HMM (Ukrytych Modeli Markowa) dla efektywnej realizacji analizy symbolicznej.

Opracowany algorytm jest niezależny od języka. Fonetyczna charakterystyka języka stanowi jedynie paramer wejściowy dla systemu rozpoznawania zdań izolowanych. Wymaga się jedynie podania zestawu głosek (fonemów) języka i ich przynależności do jednej z 7 klas: 1) dyftong (dwu-samogłoska) i normalny monoftong (samogłoska), 2) skrócony monoftong, 3) półsamogłoska, 4) głoska zwarta, 5) głoska nosowa, 6) tnąca (szczelinowa, „frykatyw") i 7) afrykat. Próbki uczące powinny być opisane fonetycznie, tzn. z wykorzystaniem jedynie tych fonemów, które zostały dostarczone na wejście algorytmu klasyfikacji mowy. Poszczególne słowa w zdaniu powinny być przedzielone znakiem specjalnym #.

Autorzy

Spis treści

1. System rozpoznawania izolowanych zdań mówionych

1.1 TRZY POZIOMY ANALIZY

Typowe podejście do komputerowej analizy zdań mówionych zakłada istnienie hierarchii trzech poziomów przetwarzania danych [RAB93, CSL00, WAL04, BEN08, KAS09]. W ramach każdego z nich wyróżniamy kroki przetwarzania:

A. analiza akustyczna

(1) analiza w dziedzinie czasu - akwizycja sygnału dźwiękowego i detekcja sygnału użytecznego mowy,

(2) analiza widmowa - przekształcenie ramek (okien) sygnału w dziedzinę częstotliwości,

(3) parametryzacja sygnału mowy - wyznaczenie wektorów (numerycznych) cech ramek (okien) sygnału,

B. analiza fonetyczna

(4) klasyfikator cech lub kwantyzacja wektorowa w terminach klas lub klastrów odpowiadających jednostkom fonetycznym mowy (tzw. trzy-fonom),

(4a) klasyfikator cech – uczenie klasyfikatora (np. geometrycznego, neuronowego, statystycznego) i wykorzystanie klasyfikatora do określania wiarygodności przynależności wektora cech do klasy fonetycznej,

(4b) kwantyzator wektorowy – klasteryzacja cech i wyznaczenie reprezentantów klastrów w trybie uczenia oraz kodowanie wektorów cech w trybie rozpoznawania,

C. analiza symboliczna

(5a) wyszukiwanie / rozpoznawanie słów kluczowych,

(5b) rozpoznawanie zdań.

Najbardziej rozpowszechnione podejścia stosowane w analizie symbolicznej mowy to: DTW („dynamic time warping” - dynamiczne marszczenie czasu) (omawiane w raporcie [KAS12]) i modelowanie stochastyczne z użyciem HMM („Hidden Markow Model” – ukryte modele Markowa) (omawiane w niniejszym raporcie).

2.2 STRUKTURA KLASYFIKACJI I ROZPOZNAWANIA SYGNAŁU MOWY

Na rys. 1.1. przedstawiono podstawowe kroki przetwarzania i typy danych w zrealizowanych algorytmach klasyfikacji i rozpoznawania zdań mówionych. Zgodnie z ogólnie przyjętą metodyką wyróżnimy kolejne etapy analizy sygnału mowy:

1. **Analiza akustyczna**
   a. **Funkcje przetwarzania wstępnego** sygnału mowy (we/wy sygnału, ewentualna filtracja sygnału, detekcja sygnału mowy (VAD – „voice activity detector”), preemfaza,
   b. **Analiza widmowa** – funkcje wyznaczania ramek sygnału i okienkowej transformaty Fouriera (STFFT),
   c. **Parametryzacja** sygnału mowy – funkcje wyznaczania cech ramek sygnału (momenty widma, cechy mel-cepstralne, cechy dodatkowe takie, jak quasi periodyczność, dolnoprzepustowość, formanty).

2. **Klasyfikacja sekwencji cech numerycznych**
   a. **Klasyfikator DTW** – dopasowanie ze sobą dwóch sekwencji wektorów cech według zasady „marszczenia czasu”, uśrednianie sekwencji wektorów cech, dopasowywanie modelu zdania do sekwencji obserwowanych cech.

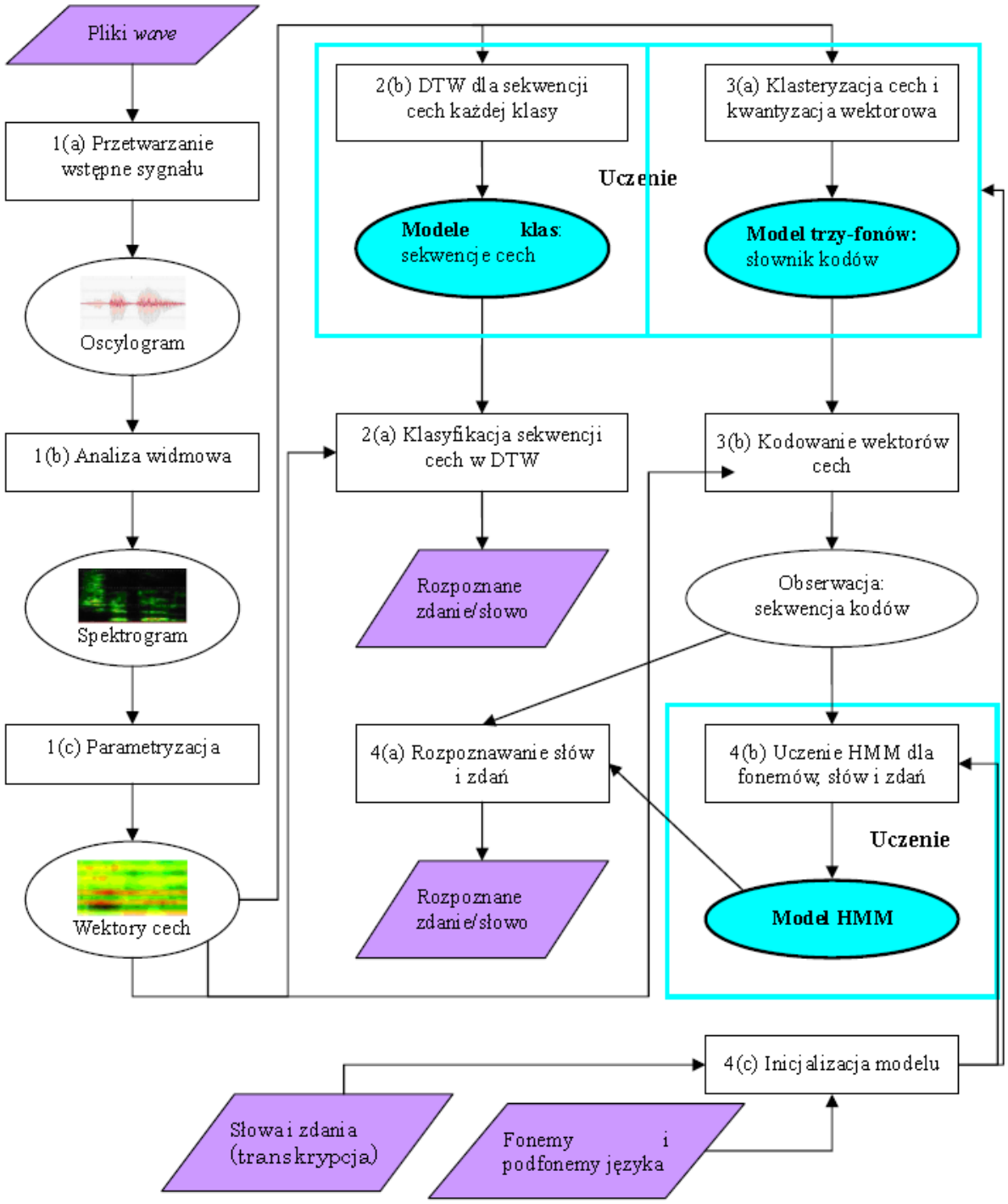
b. **Uczenie klasyfikatora** – wyznaczanie wzorcowych sekwencji cech dla słów i zdań.

3. **Koder ramek**
   a. Funkcja **KoderCech**
   Kwantyzacja wektorowa cech - analiza skupień wektorów cech w zależności od rodzaju fonemu (spółgłoska/samogłoska, dźwięczna/bezdźwięczna, ustna/nosowa, F0 i inne cechy sygnału w ramce) – sterowanie (zwiększanie liczby klas od minimalnej do maksymalnej) procesem klatesteryzacji. W drugim etapie przewidziane jest różnicowanie reprezentantów klastra zależnie od płci mówcy i częstotliwości F0.
   b. Funkcja **KoderKomend**
   Kodowanie wektora cech poprzez wybór najbliższego reprezentanta klastra. Hierarchiczny klasyfikator, który uwzględnia różne części wektora cech zależnie od kategorii głoski.

4. **Rozpoznawanie słów i zdań na poziomie symbolicznym**
   Implementacja w postaci klasy **ModelHMM.**
   a. Inicjalizacja modelu (funkcja **CreateModel**)
   Wyznaczenie 3-warstwowej struktury modelu HMM (zdania jako sekwencje słów, słowa jako sekwencje fonemów, fonemy jako sekwencje trzy-fonów) na podstawie 2 plików konfiguracyjnych zawierających listę tematów i zestaw fonemów języka. Prowadzi automatyczny podział fonemów na trzy-fony w zależności od rodzaju fonemu i określa dla modułu kodera cech minimalne i maksymalne liczby szukanych klastrów odpowiadających trzy-fonom. Wprowadza ograniczenia spodziewanych obserwacji w danym stanie modelu HMM i spodziewanego czasu trwania obserwacji.
   b. Uczenie modelu HMM (funkcja **LearnModel)**
   Algorytm uczenia 3-warstwowego modelu HMM (rozkładów przejść i wyjść) na podstawie listy tematów i wyników kodera – klasy trzy-fonów dla cech ramek sygnału. W drugim etapie możliwe jest powielanie modeli słów i zdań w zależności od rodzaju wypowiedzi – informacja, pytanie, nakaz.
      c. Zastosowanie **stochastycznych modeli HMM** w procesie poszukiwania najlepszego dopasowania modeli słów i zdań do aktualnej sekwencji jednostek fonetycznych. Funkcja **ClassifyWord -** wyszukiwanie słów kluczowych w wypowiedzi. Funkcja **ClassifySentence - k**lasyfikacja wypowiedzi w terminach zadanych sekwencji słów (tematów).

Etap 1 jest wspólny dla obu algorytmów – klasyfikacji lub rozpoznawania słów i zdań. Etap 3 ma charakter opcjonalny. Jego występowanie oznacza stosowanie w etapie 4 modeli HMM z dyskretnymi funkcjami wyjść. Przy braku etapu 3, funkcje wyjść w modelach HMM przyjmują postać mieszanin ciągłych rozkładów Gaussa.

W niniejszym raporcie przedstawiony został etap 4, czyli proces rozpoznawania w oparciu o symboliczny model HMM i jego proces uczenia. Pozostałe etapy 1-3 analizy sygnału mowy przedstawiono w poprzednim raporcie [KAS12].



Rys. 1.1 Struktura proponowanych algorytmów klasyfikacji DTW i rozpoznawania HMM sygnału mowy.

2. Słownik fonetyczny klasyfikatora HMM

Aby program klasyfikatora HMM (opisany w następnym rozdziale) mógł prawidłowo wyznaczyć strukturę modelu dla nowego zdania czy słowa, należy przekazać programowi:

1. plik z zestawem fonemów języka,
2. plik z fonetycznymi opisami zdań/słów wyrażonymi za pomocą zestawu fonemów języka, dla wszystkich próbek uczących.

Dla przykładu, dla zestawu fonemów podanych w raporcie nr 12-04 [KAS12], Autorzy przygotowali transkrypcje fonetyczne wszystkich słów występujących w zestawach uczących WAT [ADA00] i LOT [INT10].

Tab. Słownik fonetyczny wszystkich słów z zestawów zdań WAT i LOT

#=[sil]
a=[a]
aby=[a, b, I]
adres=[a, D, r, e, s]
agenta=[a, g, e, n, t, a]
akceptuje=[a, k, c, e, p, t, u, j, e]
aktualizować=[a, k, t, u, a, l, i, z, o, v, a, ć]
ale=[a, l, e]
alliance=[a, l, i, a, n, s]
and=[e, n, D]
angielskim=[a, n, g, i, e, l, s, k, i, m]
anulować=[a, n, u, l, o, v, a, ć]
anulowanie=[a, n, u, l, o, v, a, ń, e]
API=[a, p, i]
asysta=[a, s, I, s, t, a]
atrakcyjnych=[a, t, r, a, k, c, I, j, n, I, x]
audytorium=[a, u, D, I, t, o, r, i, u, m]
auta=[a, u, t, a]
bardzo=[b, a, r, Dz, o]
bagażu=[b, a, g, a, Z, u]
będą=[b, ę, D, ą]
będzie=[b, ę, dź, e]
bez=[b, e, s]
bezpieczny=[b, e, s, p, j, e, C, n, I]
bezpłatnego=[b, e, s, p, w, a, t, n, e, g, o]
bilecie=[b, i, l, e, ć, e]
bilet=[b, i, l, e, t]
biletach=[b, i, l, e, t, a, x]
biletów=[b, i, l, e, t, u, f]
biletu=[b, i, l, e, t, u]
bilety=[b, i, l, e, t, I]
biurze=[b, j, u, Z, e]
biznes=[b, i, z, n, e, s]
błąd=[b, w, o, n, D]
brak=[b, r, a, k]
być=[b, I, ć]
całego=[c, a, w, e, g, o]
cel=[c, e, l]
cena=[c, e, n, a]

cenach=[c, e, n, a, x]
cenowego=[c, e, n, o, v, e, g, o]
ceny=[c, e, n, I]
chcę=[x, c, ę]
chciałem=[x, ć, a, w, e, m]
chronione=[x, r, o, ń, o, n, e]
chwytak=[x, f, I, t, a, k]
ciąży=[ć, ą, Z, I]
ciebię=[ć, e, b, i, e]
ciepło=[ć, e, p, w, o]
cię=[ć, ę]
co=[c, o]
cookies=[k, u, k, i, s]
czasu=[C, a, s, u]
czego=[C, e, g, o]
czerwca=[C, e, r, v, c, a]
cześć=[C, e, ś, ć]
czterciestą=[C, t, e, r, dź, e, s, t, ą]
czterdzieści=[C, t, e, r, dź, e, ś, ć, i]
czternastego=[C, t, e, r, n, a, s, t, e, g, o]
czternastej=[C, t, e, r, n, a, s, t, e, j]
czternastą=[C, t, e, r, n, a, s, t, ą]
czternaście=[C, t, e, r, n, a, ś, ć, e]
cztery=[C, t, e, r, I]
czterysetną=[C, t, e, r, I, s, e, t, n, ą]
czterysta=[C, t, e, r, I, s, t, a]
czwartego=[C, f, a, r, t, e, g, o]
czwartej=[C, f, a, r, t, e, j]
czwartek=[C, f, a, r, t, e, k]
czwartą=[C, f, a, r, t, ą]
częstochowy=[C, ę, s, t, o, x, o, v, I]
czy=[C, I]
daje=[D, a, j, e]
dane=[D, a, n, e]
danych=[D, a, n, I, x]
dialogowy=[D, i, a, l, o, g, o, v, I]
dla=[D, l, a]
dlaczego=[D, l, a, C, e, g, o]
do=[D, o]
dobry=[D, o, b, r, I]
dodatkowa=[D, o, D, a, t, k, o, v, a]
dodatkowe=[D, o, D, a, t, k, o, v, e]
dokonać=[D, o, k, o, n, a, ć]
dokumentów=[D, o, k, u, m, e, n, t, u, f]
dokumenty=[D, o, k, u, m, e, n, t, I]
dostępna=[D, o, s, t, ę, p, n, a]
dotyczące=[D, o, t, I, C, ą, c, e]
dowodu=[D, o, v, o, D, u]
dowód=[D, o, v, u, D]
dół=[D, u, w]
drive=[D, r, a, j, v]
drugiego=[D, r, u, g, i, e, g, o]

drugiej=[D, r, u, g, i, e, j]
drugą=[D, r, u, g, ą]
dwa=[D, v, a]
dwadzieścia=[D, v, a, dź, e, ś, ć, a]
dwanaście=[D, v, a, n, a, ś, ć, e]
dwieście=[D, v, i, e, ś, ć, e]
dwudziestego=[D, v, u, dź, e, s, t, e, g, o]
dwudziestej=[D, v, u, dź, e, s, t, e, j]
dwudziestą=[D, v, u, dź, e, s, t, ą]
dwunastego=[D, v, u, n, a, s, t, e, g, o]
dwunastej=[D, v, u, n, a, s, t, e, j]
dwunastą=[D, v, u, n, a, s, t, ą]
dwunastu=[D, v, u, n, a, s, t, u]
dwusetną=[D, v, u, s, e, t, n, ą]
dzieci=[dź, e, ć, i]
dziecięcy=[dź, e, ć, ę, c, I]
dziecka=[dź, e, c, k, a]
dzieje=[dź, e, j, e]
dzień=[dź, e, ń]
dziesiątego=[dź, e, ś, ą, t, e, g, o]
dziesiątej=[dź, e, ś, ą, t, e, j]
dziesiątą=[dź, e, ś, ą, t, ą]
dziesięć=[dź, e, ś, ę, ć]
dziewiątego=[dź, e, v, i, ą, t, e, g, o]
dziewiątej=[dź, e, v, i, ą, t, e, j]
dziewiątą=[dź, e, v, i, ą, t, ą]
dziewiętnastego=[dź, e, v, i, ę, t, n, a, s, t, e, g, o]
dziewiętnastej=[dź, e, v, i, ę, t, n, a, s, t, e, j]
dziewiętnastą=[dź, e, v, i, ę, t, n, a, s, t, ą]
dziewiętnaście=[dź, e, v, i, ę, t, n, a, ś, ć, e]
dziewięć=[dź, e, v, j, e, ń, ć]
dziewięćdziesiąt=[dź, e, v, j, e, ń, dź, e, ś, o, n, t]
dziewięćdziesiątą=[dź, e, v, j, e, ń, dź, e, ś, o, n, t, ą]
dziewięćset=[dź, e, v, i, ę, ć, s, e, t]
dziewięćsetną=[dź, e, v, i, ę, ć, s, e, t, n, ą]
dzisiaj=[dź, i, ś, a, j]
ekonomiczną=[e, k, o, n, o, m, i, C, n, ą]
ekonomicznej=[e, k, o, n, o, m, i, C, n, e, j]
ekranie=[e, k, r, a, ń, e]
ekwipunek=[e, k, f, i, p, u, n, e, k]
elektroniczny=[e, l, e, k, t, r, o, ń, i, C, n, I]
europejskiej=[e, u, r, o, p, e, j, s, k, i, e, j]
fly=[f, l, a, j]
gdańska=[g, D, a, ń, s, k, a]
gdzie=[g, dź, e]
gdy=[g, D, I]
góra=[g, u, r, a]
grodziska=[g, r, o, dź, s, k, a]
grudnia=[g, r, u, D, ń, a]
grup=[g, r, u, p]
hej=[x, e, j]
i=[i]

ich=[i, x]
ile=[i, l, e]
inaczej=[i, n, a, C, e, j]
informacja=[i, n, f, o, r, m, a, c, j, a]
informacje=[i, n, f, o, r, m, a, c, j, e]
informacji=[i, n, f, o, r, m, a, c, j, i]
inna=[i, n, n, a]
inną=[i, n, n, ą]
inne=[i, n, n, e]
innego=[i, n, n, e, g, o]
inny=[i, n, n, I]
innych=[i, n, n, I, x]
innym=[i, n, n, I, m]
interesują=[i, n, t, e, r, e, s, u, j, ą]
interesuje=[i, n, t, e, r, e, s, u, j, e]
internecie=[i, n, t, e, r, n, e, ć, e]
internet=[i, n, t, e, r, n, e, t]
internetowej=[i, n, t, e, r, n, e, t, o, v, e, j]
ja=[j, a]
jadę=[j, a, D, ę]
jeden=[j, e, D, e, n]
jedenastą=[j, e, D, e, n, a, s, t, ą]
jedenastu=[j, e, D, e, n, a, s, t, u]
jedenaście=[j, e, D, e, n, a, ś, ć, e]
jedną=[j, e, D, n, ą]
jedynastego=[j, e, D, I, n, a, s, t, e, g, o]
jedynastej=[j, e, D, I, n, a, s, t, e, j]
jak=[j, a, k]
jaka=[j, a, k, a]
jaki=[j, a, k, i]
jakich=[j, a, k, i, x]
jakie=[j, a, k, i, e]
jakim=[j, a, k, i, m]
jej=[j, e, j]
jest=[j, e, s, t]
języku=[j, ę, z, I, k, u]
jutro=[j, u, t, r, o]
jutrze=[j, u, t, S, e]
kalendarza=[k, a, l, e, n, D, a, Z, a]
kanada=[k, a, n, a, D, a]
kanady=[k, a, n, a, D, I]
kartą=[k, a, r, t, ą]
kąt=[k, ą, t]
kiedy=[k, i, e, D, I]
klasą=[k, l, a, s, ą]
klasie=[k, l, a, ś, e]
kobiet=[k, o, b, i, e, t]
kod=[k, o, D]
kol=[k, o, l]
kolejnym=[k, o, l, e, j, n, I, m]
kont=[k, o, n, t]
kontakt=[k, o, n, t, a, k, t]

kopii=[k, o, p, j, i]
korzyści=[k, o, Z, I, ś, ć, i]
kraj=[k, r, a, j]
krajowe=[k, r, a, j, o, v, e]
krajowy=[k, r, a, j, o, v, I]
krajowych=[k, r, a, j, o, v, I, x]
kraju=[k, r, a, j, u]
krakowa=[k, r, a, k, o, v, a]
ktoś=[k, t, o, ś]
której=[k, t, u, r, e, j]
kupić=[k, u, p, i, ć]
kwietnia=[k, v, i, e, t, n, i, a]
lat=[l, a, t]
lecę=[l, e, c, ę]
lewo=[l, e, v, o]
limit=[l, i, m, i, t]
limity=[l, i, m, i, t, I]
lipca=[l, i, p, c, a]
listopada=[l, i, s, t, o, p, a, D, a]
litwina=[l, i, t, v, i, n, a]
logować=[l, o, g, o, v, a, ć]
logowaniu=[l, o, g, o, v, a, ń, u]
lojalnościowe=[l, o, j, a, l, n, o, ś, ć, o, v, e]
lotach=[l, o, t, a, x]
lotnisku=[l, o, t, ń, i, s, k, u]
lotów=[l, o, t, u, f]
lotu=[l, o, t, u]
loty=[l, o, t, I]
lub=[l, u, p]
lutego=[l, u, t, e, g, o]
maja=[m, a, j, a]
mam=[m, a, m]
marca=[m, a, r, c, a]
maestro=[m, a, e, s, t, r, o]
mi=[m, i]
miejsc=[m, i, j, e, j, s, c]
miejsca=[m, i, j, e, j, s, c, a]
międzynarodowe=[m, i, ę, Dz, I, n, a, r, o, D, o, v, e]
międzynarodowy=[m, i, ę, Dz, I, n, a, r, o, D, o, v, I]
mil=[m, i, l]
mile=[m, i, l, e]
miles=[m, a, j, l, s]
miło=[m, i, w, o]
minut=[m, i, n, u, t]
mnie=[m, ń, e]
mogą=[m, o, g, ą]
mogę=[m, o, g, e]
moja=[m, o, j, a]
moje=[m, o, j, e]
more=[m, o, r]
może=[m, o, Z, e]
możliwe=[m, o, Z, l, i, v, e]

można=[m, o, Z, n, a]
mój=[m, u, j]
muszę=[m, u, S, ę]
na=[n, a]
nadbagaż=[n, a, D, b, a, g, a, Z]
najbliższy=[n, a, j, b, l, i, Z, S, I]
należy=[n, a, l, e, Z, I]
naliczane=[n, a, l, i, C, a, n, e]
napoje=[n, a, p, o, j, e]
narciarski=[n, a, r, ć, a, r, s, k, i]
narciarskiego=[n, a, r, ć, a, r, s, k, i, e, g, o]
następny=[n, a, s, t, ę, p, n, I]
nazwisko=[n, a, z, v, i, s, k, o]
nic=[ń, i, c]
nie=[ń, e]
niedzielę=[ń, i, e, dź, e, l, e]
niemowlęciem=[ń, e, m, o, v, l, ę, ć, e, m]
niewykorzystany=[ń, e, v, I, k, o, Z, I, s, t, a, n, I]
niż=[ń, i, Z]
nocy=[n, o, c, I]
numer=[n, u, m, e, r]
o=[o]
obsługi=[o, p, s, w, u, g, i]
obsługuje=[o, p, s, w, u, g, u, j, e]
ochrony=[o, x, r, o, n, I]
od=[o, D]
odbiór=[o, D, b, i, u, r]
odebrać=[o, D, e, b, r, a, ć]
odjeżdża=[o, D, j, e, Z, dZ, a]
odlotem=[o, D, l, o, t, e, m]
odpowiedzialność=[o, D, p, o, v, i, e, dź, a, l, n, o, ś, ć]
odprawa=[o, D, p, r, a, v, a]
odprawić=[o, D, p, r, a, v, i, ć]
odprawie=[o, D, p, r, a, v, i, e]
odprawy=[o, D, p, r, a, v, I]
ofercie=[o, f, e, r, ć, e]
oferowane=[o, f, e, r, o, v, a, n, e]
ograniczenia=[o, g, r, a, ń, i, C, e, ń, a]
opieka=[o, p, i, e, k, a]
opieki=[o, p, i, e, k, i]
opłata=[o, p, w, a, t, a]
opłaty=[o, p, w, a, t, I]
organom=[o, r, g, a, n, o, m]
os=[o, s]
osiem=[o, ś, e, m]
osiemdziesiąt=[o, ś, e, m, dź, e, ś, o, n, t]
osiemdziesiątą=[o, ś, e, m, dź, e, ś, o, n, t, ą]
osiemnastego=[o, ś, e, m, n, a, s, t, e, g, o]
osiemnastej=[o, ś, e, m, n, a, s, t, e, j]
osiemnastą=[o, ś, e, m, n, a, s, t, ą]
osiemnaście=[o, ś, e, m, n, a, ś, ć, e]
osiemset=[o, ś, e, m, s, e, t]

osiemsetną=[o, ś, e, m, s, e, t, n, ą]
osobę=[o, s, o, b, ę]
osobistego=[o, s, o, b, i, s, t, e, g, o]
osobisty=[o, s, o, b, i, s, t, I]
osobowe=[o, s, o, b, o, v, e]
osobowych=[o, s, o, b, o, v, I, x]
ósmego=[u, s, m, e, g, o]
ósmej=[u, s, m, e, j]
ósmą=[u, s, m, ą]
oś=[o, ś]
partnerami=[p, a, r, t, n, e, r, a, m, i]
pasażerów=[p, a, s, a, Z, e, r, u, f]
paszport=[p, a, S, p, o, r, t]
października=[p, a, ź, dź, e, r, n, i, k, a]
personalne=[p, e, r, s, o, n, a, l, n, e]
pierwszego=[p, i, e, r, v, S, e, g, o]
pierwszej=[p, i, e, r, v, S, e, j]
pierwszy=[p, i, e, r, v, S, I]
piątego=[p, i, o, n, t, e, g, o]
piątej=[p, i, o, n, t, e, j]
piątek=[p, i, o, n, t, e, k]
piątą=[p, i, o, n, t, ą]
pieniędzy=[p, i, e, ń, ę, Dz, I]
pięciu=[p, i, ę, ć, u]
piętnastego=[p, i, ę, t, n, a, s, t, e, g, o]
piętnastej=[p, i, ę, t, n, a, s, t, e, j]
piętnastą=[p, i, ę, t, n, a, s, t, ą]
piętnaście=[p, i, ę, t, n, a, ś, ć, e]
pięć=[p, j, e, ń, ć]
pięćdziesiąt=[p, j, e, ń, ć, dź, e, ś, o, n, t]
pięćdziesiątą=[p, j, e, ń, ć, dź, e, ś, o, n, t, ą]
pięćset=[p, j, e, ń, ć, s, e, t]
pięćsetną=[p, j, e, ń, ć, s, e, t, n, ą]
PIN=[p, i, n]
plików=[p, l, i, k, u, f]
płacę=[p, w, a, c, ę]
płacić=[p, w, a, ć, i, ć]
PNR=[p, e, e, n, e, r]
po=[p, o]
pociąg=[p, o, ć, ą, g]
podać=[p, o, D, a, ć]
podczas=[p, o, D, C, a, s]
podręcznym=[p, o, D, r, ę, C, n, I, m]
podróż=[p, o, D, r, u, Z]
pokładzie=[p, o, k, w, a, dź, e]
polsce=[p, o, l, s, c, e]
poniedziałek=[p, o, n, i, e, dź, a, w, e, k]
poniżej=[p, o, ń, i, Z, e, j]
posiłki=[p, o, ś, i, w, k, i]
potrzebne=[p, o, t, S, e, b, n, e]
potrzebuję=[p, o, t, S, e, b, u, j, ę]
potwierdzenie=[p, o, t, f, i, e, r, Dz, e, ń, e]

potwierdzeniem=[p, o, t, f, i, e, r, Dz, e, ń, e, m]
powiedz=[p, o, v, i, e, Dz]
powrót=[p, o, v, r, u, t]
pozdrawiam=[p, o, z, D, r, a, v, i, a, m]
południu=[p, o, w, u, D, n, i, u]
prawne=[p, r, a, v, n, e]
prawo=[p, r, a, v, o]
problem=[p, r, o, b, l, e, m]
profilu=[p, r, o, f, i, l, u]
program=[p, r, o, g, r, a, m]
programie=[p, r, o, g, r, a, m, i, e]
programu=[p, r, o, g, r, a, m, u]
programy=[p, r, o, g, r, a, m, I]
próbuję=[p, r, u, b, u, j, e]
przebiega=[p, Z, e, b, i, e, g, a]
przechowywania=[p, S, e, x, o, v, I, v, a, ń, a]
przed=[p, Z, e, D]
przeglądarka=[p, Z, e, g, l, o, n, D, a, r, k, a]
przekazywane=[p, Z, e, k, a, z, I, v, a, n, e]
przekazywania=[p, Z, e, k, a, z, I, v, a, ń, a]
przekazywaniu=[p, Z, e, k, a, z, I, v, a, ń, u]
przekazywanych=[p, Z, e, k, a, z, I, v, a, n, I, x]
przelewem=[p, Z, e, l, e, v, e, m]
przeloty=[p, Z, e, l, o, t, I]
przewieźć=[p, Z, e, v, i, e, ś, ć]
przewozić=[p, Z, e, v, o, ź, i, ć]
przewozu=[p, Z, e, v, o, z, u]
przewóz=[p, Z, e, v, u, s]
przez=[p, Z, e, s]
przy=[p, Z, I]
przyjedzie=[p, Z, I, j, e, dź, e]
puść=[p, u, ś, ć]
ramach=[r, a, m, a, x]
ramy=[r, a, m, I]
rano=[r, a, n, o]
rejestracja=[r, e, j, e, s, t, r, a, c, j, a]
rejsu=[r, e, j, s, u]
rezerwacja=[r, e, z, e, r, v, a, c, j, a]
rezerwacjach=[r, e, z, e, r, v, a, c, j, a, x]
rezerwacje=[r, e, z, e, r, v, a, c, j, e]
rezerwację=[r, e, z, e, r, v, a, c, j, ę]
rezerwacji=[r, e, z, e, r, v, a, c, j, i]
rezerwować=[r, e, z, e, r, v, o, v, a, ć]
rodzaj=[r, o, Dz, a, j]
rower=[r, o, v, e, r]
rozpocząć=[r, o, s, p, o, C, ą, ć]
ryszki=[r, I, S, k, i]
salę=[s, a, l, ę]
samolocie=[s, a, m, o, l, o, ć, e]
samolotem=[s, a, m, o, l, o, t, e, m]
są=[s, ą]
senter=[s, e, n, t, e, r]

serca=[s, e, r, c, a]
serdecznie=[s, e, r, D, e, C, ń, e]
setną=[s, e, t, n, ą]
siebie=[ś, e, b, j, e]
siedem=[ś, e, D, e, m]
siedemdziesiąt=[ś, e, D, e, m, dź, e, ś, o, n, t]
siedemdziesiątą=[ś, e, D, e, m, dź, e, ś, o, n, t, ą]
siedemnastego=[ś, e, D, e, m, n, a, s, t, e, g, o]
siedemnastej=[ś, e, D, e, m, n, a, s, t, e, j]
siedemnastą=[ś, e, D, e, m, n, a, s, t, ą]
siedemnastu=[ś, e, D, e, m, n, a, s, t, u]
siedemnaście=[ś, e, D, e, m, n, a, ś, ć, e]
siedemset=[ś, e, D, e, m, s, e, t]
siedemsetną=[ś, e, D, e, m, s, e, t, n, ą]
sierpnia=[ś, e, r, p, ń, a]
się=[ś, ę]
siódmego=[ś, u, D, m, e, g, o]
siódmej=[ś, u, D, m, e, j]
siódmą=[ś, u, D, m, ą]
skład=[s, k, w, a, D]
skorzystać=[s, k, o, Z, I, s, t, a, ć]
skorygować=[s, k, o, r, I, g, o, v, a, ć]
sobą=[s, o, b, ą]
sobotę=[s, o, b, o, t, e]
specjalna=[s, p, e, c, j, a, l, n, a]
sportowy=[s, p, o, r, t, o, v, I]
sposób=[s, p, o, s, u, p]
sprawie=[s, p, r, a, v, i, e]
sprzedaży=[s, p, Z, e, D, a, Z, I]
sprzęt=[s, p, Z, ę, t]
sprzętu=[s, p, Z, ę, t, u]
SSL=[e, s, e, s, e, l]
stać=[s, t, a, ć]
stanów=[s, t, a, n, u, f]
stany=[s, t, a, n, I]
star=[s, t, a, r]
start=[s, t, a, r, t]
statusowych=[s, t, a, t, u, s, o, v, I, x]
stawek=[s, t, a, v, e, k]
stawka=[s, t, a, v, k, a]
stawki=[s, t, a, v, k, i]
sto=[s, t, o]
stop=[s, t, o, p]
stracił=[s, t, r, a, ć, i, w]
stronie=[s, t, r, o, ń, e]
stycznia=[s, t, I, C, ń, a]
swoje=[s, f, o, j, e]
system=[s, I, s, t, e, m]
szczególnie=[S, C, e, g, u, l, ń, e]
szesnastego=[S, e, s, n, a, s, t, e, g, o]
szesnastej=[S, e, s, n, a, s, t, e, j]
szesnastą=[S, e, s, n, a, s, t, ą]

szesnaście=[S, e, s, n, a, ś, ć, e]
sześć=[S, e, ś, ć]
sześćdziesiąt=[S, e, ś, ć, dź, e, ś, o, n, t]
sześćdziesiątą=[S, e, ś, ć, dź, e, ś, o, n, t, ą]
sześćset=[S, e, ś, ć, s, e, t]
sześćsetną=[S, e, ś, ć, s, e, t, n, ą]
szóstego=[S, u, s, t, e, g, o]
szóstej=[S, u, s, t, e, j]
szóstą=[S, u, s, t, ą]
środę=[ś, r, o, D, ę]
świat=[ś, f, i, a, t]
tak=[t, a, k]
taryfy=[t, a, r, I, f, I]
te=[t, e]
telefon=[t, e, l, e, f, o, n]
telefonu=[t, e, l, e, f, o, n, u]
telefoniczna=[t, e, l, e, f, o, ń, i, C, n, a]
telefonicznej=[t, e, l, e, f, o, ń, i, C, n, e, j]
telefonicznie=[t, e, l, e, f, o, ń, i, C, ń, e]
temat=[t, e, m, a, t]
to=[t, o]
tradycyjnej=[t, r, a, D, I, c, I, j, n, e, j]
trzeba=[t, S, e, b, a]
trzeciego=[t, S, e, ć, e, g, o]
trzeciej=[t, S, e, ć, e, j]
trzecią=[t, S, e, ć, ą]
trzy=[t, S, I]
trzydziestego=[t, S, I, dź, e, s, t, e, g, o]
trzydziestą=[t, S, I, dź, e, s, t, ą]
trzydzieści=[t, S, I, dź, e, ś, ć, i]
trzynastego=[t, S, I, n, a, s, t, e, g, o]
trzynastej=[t, S, I, n, a, s, t, e, j]
trzynastą=[t, S, I, n, a, s, t, ą]
trzynaście=[t, S, I, n, a, ś, ć, e]
trzysetną=[t, S, I, s, e, t, n, ą]
trzysta=[t, S, I, s, t, a]
tylko=[t, I, l, k, o]
u=[u]
uczestnikiem=[u, C, e, s, t, ń, i, k, i, e, m]
udających=[u, D, a, j, o, n, c, I, x]
udostępniane=[u, D, o, s, t, ę, p, ń, a, n, e]
udostępnianie=[u, D, o, s, t, ę, p, ń, a, ń, e]
udostępnione=[u, D, o, s, t, ę, p, ń, o, n, e]
unii=[u, ń, i]
usługi=[u, s, w, u, g, i]
użyć=[u, Z, I, ć]
użytkownika=[u, Z, I, t, k, o, v, ń, i, k, a]
visa=[v, i, z, a]
w=[v]
warszawa=[v, a, r, S, a, v, a]
warszawy=[v, a, r, S, a, v, I]
warszawę=[v, a, r, S, a, v, e]

warunki=[v, a, r, u, n, k, i]
was=[v, a, s]
ważność=[v, a, Z, n, o, ś, ć]
wchodzi=[v, x, o, dź, i]
we=[v, e]
wewnątrz=[v, e, v, n, ą, t, S]
wgląd=[v, g, l, o, n, D]
wieczorem=[v, j, e, C, o, r, e, m]
wiedzieć=[v, j, e, dź, e, ć]
więcej=[v, j, ę, c, e, j]
witaj=[v, i, t, a, j]
witam=[v, i, t, a, m]
wliczony=[v, l, i, C, o, n, I]
władzom=[v, w, a, Dz, o, m]
własną=[v, w, a, s, n, ą]
wolnego=[v, o, l, n, e, g, o]
wolno=[v, o, l, n, o]
wózek=[v, u, z, e, k]
wraz=[v, r, a, s]
wrocławia=[v, r, o, c, w, a, v, i, a]
września=[v, Z, e, ś, ń, a]
współpracuje=[v, s, p, u, w, p, r, a, c, u, j, e]
wtorek=[v, t, o, r, e, k]
wydanie=[v, I, D, a, ń, e]
wymagane=[v, I, m, a, g, a, n, e]
wynajmie=[v, I, n, a, j, m, i, e]
wystarczy=[v, I, s, t, a, r, C, I]
wyświetla=[v, I, ś, f, i, e, t, l, a]
z=[z]
za=[z, a]
zabezpieczone=[z, a, b, e, z, p, i, e, C, o, n, e]
zabrać=[z, a, b, r, a, ć]
zajmij=[z, a, j, m, i, j]
zakup=[z, a, k, u, p]
zakupem=[z, a, k, u, p, e, m]
zalogować=[z, a, l, o, g, o, v, a, ć]
zamieszkania=[z, a, m, i, e, S, k, a, ń, a]
zapisz=[z, a, p, i, S]
zapłacić=[z, a, p, w, a, ć, i, ć]
zarezerwować=[z, a, r, e, z, e, r, v, o, v, a, ć]
zarezerwuj=[z, a, r, e, z, e, r, v, u, j]
zawsze=[z, a, v, S, e]
zbiera=[z, b, i, e, r, a]
zbieranie=[z, b, i, e, r, a, ń, e]
zdobycie=[z, D, o, b, I, ć, e]
ze=[z, e]
zero=[z, e, r, o]
zgłoszenia=[z, g, w, o, S, e, ń, a]
zjednoczone=[z, j, e, D, n, o, C, o, n, e]
zjednoczonych=[z, j, e, D, n, o, C, o, n, I, x]
złap=[z, w, a, p]
zmiana=[z, m, i, a, n, a]

zmienić=[z, m, i, e, ń, i, ć]
znajdę=[z, n, a, j, D, e]
zniżki=[z, ń, i, Z, k, i]
zrobić=[z, r, o, b, i, ć]
zwierzę=[z, v, i, e, Z, ę]
zwrot=[z, v, r, o, t]

3. Model HMM dla pojedynczych zdań

Opisano algorytmy:
1. do wyznaczania struktury modeli HMM na podstawie plików konfiguracyjnych z opisem klasyfikowanych zdań i słów oraz podstawową klasyfikacją fonemów danego języka,
2. uczenia parametrów modelu HMM na podstawie próbek uczących o postaci kodów cech i
3. rozpoznawania sekwencji kodów w termiach modeli HMM dla zdań i słów.

Program do rozpoznawania zdań z modelem HMM zrealizowano w **Javie** w środowisku **NetBeans**. Jest on przeznaczony do inicjalizacji modelu, uczenia parametrów dyskretnego modelu HMM i rozpoznawania słów kluczowych oraz całych zdań w sygnale mowy. Dane wejściowe są postaci przetworzonego sygnału mowy – sekwencji kodow jednostek fonetycznych.

3.1 INICJALIZACJA STRUKTURY MODELU

Fonetyczny model komendy składa się z sekwencji jednostek fonetycznych, zwanych pod-fonemami, przy uwzględnieniu możliwości występowania zmian w aktualnej wypowiedzi, polegających na:

- wydłużeniu czasu trwania danego pod-fonemu do kilku kolejnych okien,
- występowaniu pojedynczych przekłamań na inne „zbliżone" pod-fonemy,
- pomijaniu pojedynczych pod-fonemów występujących w modelu.

Odpowiednią formą reprezentacji takich sytuacji jest *lewo-prawy* Ukryty Model Markowa (HMM), który w zwartej formie przedstawia różne transkrypcje fonetyczne wymowy słowa [HTK06], [WYD09].

Ukryte Modele Markowa

Ukryte Modele Markowa należą do grupy łańcuchów (sekwencji) stochastycznych Markowa. Można je opisać w następujący sposób:

Niech S będzie skończonym i niepustym zbiorem stanów. Pewne stany s_{start} oraz s_{stop} należące do S są oznaczone jako stan początkowy i końcowy. Łańcuch Markowa jest zadany przez macierz przejść A, która dla dowolnych dwóch stanów s_k i s_l należących do S opisuje prawdopodobieństwo przejścia ze stanu s_k do s_l. Dla każdego stanu należącego do S, za wyjątkiem s_{stop}, musi być spełniony dodatkowo warunek, by suma prawdopodobieństw wyjść z tego stanu była równa 1. Jest to więc pewien graf skierowany, w którym wagi na przejściach oznaczają prawdopodobieństwo przejścia do stanu następnego.

Łańcuch Markowa opisuje układ, który w każdym momencie znajduje się tylko w jednym ze stanów należących do S. Układ ten jest obserwowany w dyskretnych chwilach czasowych $t = 0, 1, 2, \ldots T$. Zakłada się, że na początku układ znajduje się w stanie początkowym s_{start}, natomiast na końcu przechodzi do stanu końcowego s_{stop}.

Cechą wyróżniającą Ukryte Modele Markowa spośród innych łańcuchów Markowa jest to, że prawdopodobieństwo przejścia do następnego stanu, w dowolnej chwili, zależy tylko od stanu obecnego i nie zależy od historii (sposobu w jaki proces doszedł do obecnego stanu) (tzw. model Markowa 1-szego rzędu). Jeśli w chwili t układ znajduje się w stanie s_k, to w chwili $t+1$ może on przejść do stanu s_l z prawdopodobieństwem opisanym przez macierz przejść: $A(k,l)$. Nie jest istotne w jakim stanie był proces w chwilach $t-1$, $t-2$, $t-3$, … .[1]

[1] Przedstawiony tu opis jest bardzo uproszczony i został tu umieszczony w celu zapoznania czytelnika z podstawowymi pojęciami wykorzystywanymi w pracy.

Z drugiej strony Ukryte Modele Markowa (HMM, od „Hidden Markov Models") rozszerzają definicję łańcucha Markowa, gdyż łączą ze sobą dwie sekwencji (procesy) stochastyczne.

Niech *Delta* będzie alfabetem. Zakłada się, że Ukryte Modele Markowa są łańcuchami Markowa mogącymi komunikować się z otoczeniem za pomocą emitowania liter z alfabetu *Delta*. Tak więc mając dany HMM będący w stanie s_k, obserwujemy symbol *o* należący do *Delta* z prawdopodobieństwem $B(o, k)$, a układ przechodzi do stanu s_l z prawdopodobieństwem $A(k, l)$.

Przyjmuje się, że to co daje się obserwować to symbole generowane podczas przechodzenia przez Model Markowa w kolejnych chwilach czasowych. Stany są w tym czasie ukryte (stąd nazwa Ukryte Modele Markowa).

W pracy będą przyjmowane następujące oznaczenia:

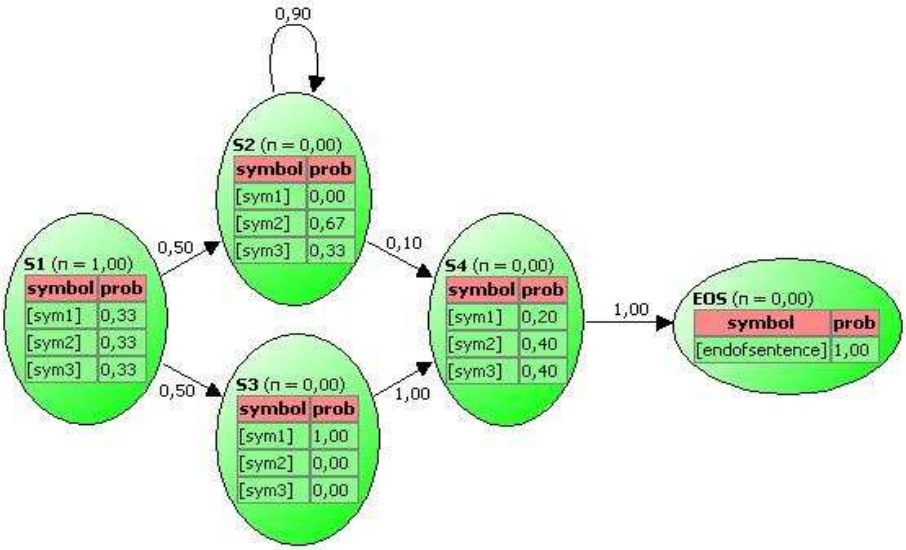
- $\{o_0, o_1, o_2, \ldots, o_T\}$ – sekwencja obserwacji w kolejnych chwilach czasowych,
- **Delta –** alfabet zawierający symbole generowane przez model ($o_0, o_1, o_2, \ldots, o_T$ należą do **Delta**),
- **T** = |O| - długość sekwencji obserwacji,
- **S** – $\{s_0, s_1, s_2, \ldots, s_N\}$ – zbiór stanów HMM,
- **N** = |S| - ilość stanów modelu,
- **Pi** = $\{\pi_0, \pi_1, \pi_2, \ldots, \pi_N\}$ – prawdopodobieństwa przejść ze stanu początkowego s_{start} do stanu o numerze 1, 2, 3, …, N,
- **Sigma** = $\{\sigma_0, \sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_N\}$ – prawdopodobieństwa przejść do stanu końcowego s_{stop} ze stanu o numerze 1, 2, 3, …, N,
- **EOS** – stan końcowy s_{stop} ,
- **A** – macierz opisująca prawdopodobieństwo $A(k, l)$ przejścia ze stanu s_k do s_l,
- **B** – macierz opisująca prawdopodobieństwo $B(o,k)$ wyemitowania obserwacji *o* w stanie s_k.

Przykład:
Ukryty Model Markowa zawierający 4 stany przedstawiono na rysunku 3.1. Dla tego HMM mamy następujący zbiór parametrów:

- **S** = {S1, S2, S3, S4} – stany modelu (stan początkowy jest tutaj niewidoczny, stan końcowy EOS),
- **Delta** = {sym1, sym2, sym3} – zbiór symboli, które możemy obserwować.

$$\mathbf{Pi}=\begin{bmatrix}1.00\\0.00\\0.00\\0.00\end{bmatrix}, Sigma=\begin{bmatrix}0.00\\0.00\\0.00\\1.00\end{bmatrix}, \mathbf{A}=\begin{bmatrix}0.00 & 0.50 & 0.50 & 0.00\\0.00 & 0.90 & 0.00 & 0.10\\0.00 & 0.00 & 0.00 & 1.00\\0.00 & 0.00 & 0.00 & 0.00\end{bmatrix}, \mathbf{B}=\begin{bmatrix}0.33 & 0.33 & 0.33\\0.00 & 0.67 & 0.33\\1.00 & 0.00 & 0.00\\0.20 & 0.40 & 0.40\end{bmatrix}$$

W przykładzie pominięto stan początkowy. Prawdopodobieństwo przejścia ze stanu początkowego do któregoś ze stanów modelu określone jest przez macierz **Pi**. Macierz **Pi** można też rozumieć jako rozkład przyporządkowujący stanom prawdopodobieństwo rozpoczęcia w nich sekwencji obserwacji. Prawdopodobieństwo przejścia do stanu *EOS* ze stanu s_k jest opisane w macierzy *Sigma* i równe jest prawdopodobieństwu zakończenia sekwencji obserwacji w stanie s_k.



Rysunek 3.1 - Przykład Ukrytego Modelu Markowa

Fonetyczne Ukryte Modele Markowa

Obserwacje
W procesie rozpoznawania mowy w dyskretnych chwilach czasowych obserwujemy sekwencję kolejnych klas fonetycznych. Odpowiada ona sekwencji obserwacji generowanej podczas przechodzenia przez Ukryty Model Markowa. Zgodnie z przyjętą konwencją:

Delta – alfabet HMM. Zbiór wszystkich ponumerowanych klas fonetycznych (klasteryzacja cech),
O – $[o_0, o_1, o_2, \ldots, o_T]$ – sekwencja klas fonetycznych (kwantyzacja cech).

Struktura
Idealny HMM, który z prawdopodobieństwem równym *1* generuje zadaną sekwencję klas fonetycznych, powinien posiadać dla każdej klasy z sekwencji odpowiadający jej stan. W idealnym modelu w dowolnej chwili *t* prawdopodobieństwo wygenerowania w stanie s_t obserwacji o_t byłaby równa 1. Tak samo prawdopodobieństwo przejścia do stanu s_{t+1} też byłoby równe 1. Idealny model ma jednak pewną wadę – generuje zawsze tę samą sekwencję.

Aby zmienić strukturę modelu na bardziej elastyczną weźmy pod uwagę, że ta sama klasa może powtarzać się w kilu kolejnych chwilach. Zamienimy więc odpowiadającą tym klasom sekwencję stanów na jeden stan z pętlą. Zyskujemy pewną elastyczność i mniejszą ilość stanów. Prawdopodobieństwo generowania wejściowej sekwencji się jednak zmniejsza.

Dalsza modyfikacja to dopuszczenie przejść omijających niektóre stany. W ten sposób modelujemy sekwencje, w których pewne klasy zostały zgubione.

Model Pod-Fonemowy
W praktyce nie da się prosto zbudować HMM na podstawie sekwencji obserwacji. Posiadając natomiast zapis fonetyczny słowa i znając reguły podziału fonemów na pod-fonemy możemy zbudować HMM zbliżony do opisanego wcześniej. Pod-fonemowy HMM będzie jednak posiadał innym alfabetem wyjściowym. Alfabet ten będzie się składał ze wszystkich pod-fonemów występujących w języku.

Gamma – alfabet pod-fonemów
B$^{\textbf{fonem}}$ **–** macierz opisująca prawdopodobieństwo $B^{fonem}(f_i , s_k)$ wyemitowania pod-fonemu f_i w stanie s_k.

Pod-Fonemy i klasy fonetyczne
Aby połączyć model pod-fonemowy z sekwencją klas fonetycznych musimy określić odpowiednie mapowanie pod-fonemów na klasy.

Delta – alfabet klas fonetycznych,
B$^{\textbf{klasa}}$ **–** macierz opisująca prawdopodobieństwo $B^{klasa}(k_p \mid f_i)$ wyemitowania klasy fonetycznej k_p pod warunkiem wyemitowania pod-fonemu f_i .

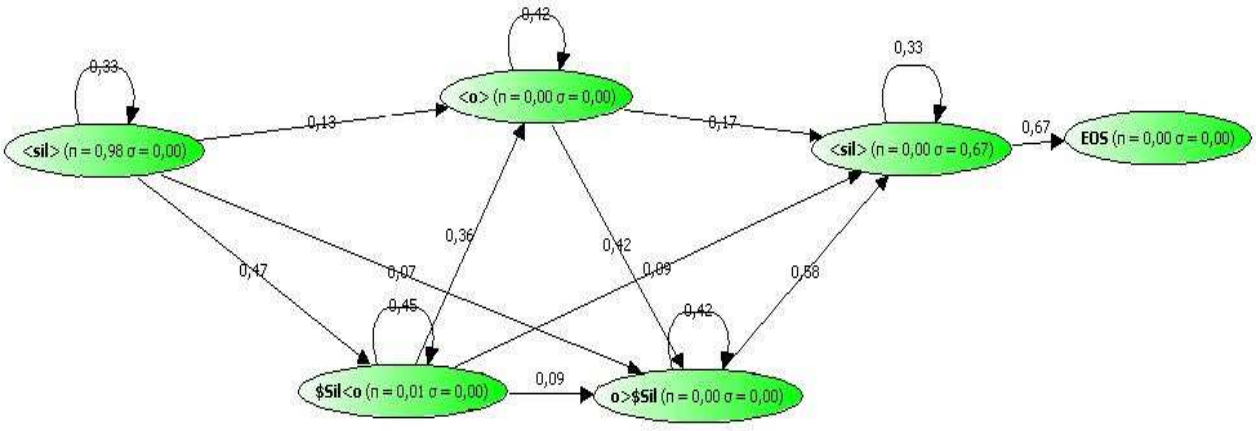
Macierze **B**$^{\textbf{fonem}}$ i **B**$^{\textbf{klasa}}$ pozwalają wyznaczyć prawdopodobieństwo wyemitowania klasy fonetycznej w dowolnym stanie modelu:

$$B = \sum_{i=0}^{M} B^{fonem}(f_i, s_k) * B^{klasa}(k_p | f_i)$$

Gdzie B jest prawdopodobieństwem wyemitowania klasy fonetycznej k_p w stanie s_k, natomiast M oznacza ilość pod-fonemów .

Nazewnictwo stanów
Zakładając, że poza szczególnymi przypadkami, w każdym stanie HMM będziemy emitować jeden typ pod-fonemu (liczba stanów wynika z podziału słowa na pod-fonemy) zastosujemy nazewnictwo stanów odpowiadające nazwie pod-fonemu, który może być w nich generowany. Jeśli w stanie możemy emitować więcej niż jeden typ pod-fonemu, to nazwa stanu będzie się składać z oddzielonych przecinkami nazw pod-fonemów.

<u>Przykład:</u>
Słowo: o
Zapis fonetyczny[2]: /sil/, /o/, /sil/
Rozbicie na pod-fonemy: <sil> ; &Sil<o ; <o> ; o>&Sil ; <sil>
Wynikowy model jest pokazany na rysunku 3.2.



Rysunek 3.2 – HMM pod-fonemowy

[2] Na początku i końcu dodano fonem ciszy. Stanowi on kontekst dla fonemu *o*.

Generowanie Ukrytych Modeli Markowa

Do wygenerowania fonetycznego HMM powinniśmy mieć określone:

- zdanie,
- zapis fonetyczny słów w zdaniu,
- reguły podziału fonemów na pod-fonemy.

Na podstawie tych informacji generujemy 3-poziomowy HMM składający się z:

1. modelu zdania (stany modelu odpowiadają słowom),
2. modelu słów (stany modelu odpowiadają fonemom),
3. modelu fonemów (stany modelu odpowiadają pod-fonemom).

Zaczynamy od zbudowania HMM opisującego zdanie. Następnie rozbijamy HMM zdania na HMM słów. Zamieniamy w tym procesie stany reprezentujące słowa na pod-modele z fonemami. Dodatkowo wykorzystujemy macierze **Pi** i **Sigma**, by wyznaczyć prawdopodobieństwa przejść **A** pomiędzy pod-modelami. Podobnie rozbijamy HMM słów zawierające fonemy na właściwy HMM pod-fonemowy.

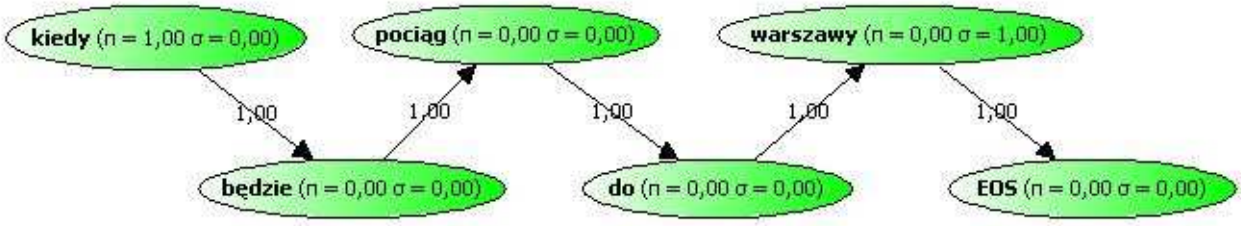
Poziom słów

W pierwszym kroku konstruujemy prosty HMM generujący słowa w zdaniu. Kierujemy się przy tym następującymi regułami[3]:

- Wartość **Pi** dla stanu odpowiadającemu pierwszemu słowu wynosi 1. W p. p. *0*,
- Wartość *Sigma* dla stanu odpowiadającemu ostatniemu słowu wynosi 1. W p. p. *0*,
- Prawdopodobieństwo przejścia pomiędzy stanami odpowiadającymi kolejnym słowom wynosi 1. W p. p. *0*,
- Prawdopodobieństwo wyemitowania słowa w stanie przypisanym do słowa wynosi 1. W p. p. *0*.

Przykład:

Zdanie: kiedy będzie pociąg do Warszawy
Wynik: na rysunku 3.3



Rysunek 3.3 –HMM słów

Poziom fonemów

W kolejnym kroku rozbijamy stany odpowiadające słowom na fonemy. Będą one odzwierciedlały zapisy fonetyczne słów zdefiniowany w słowniku fonetycznym.

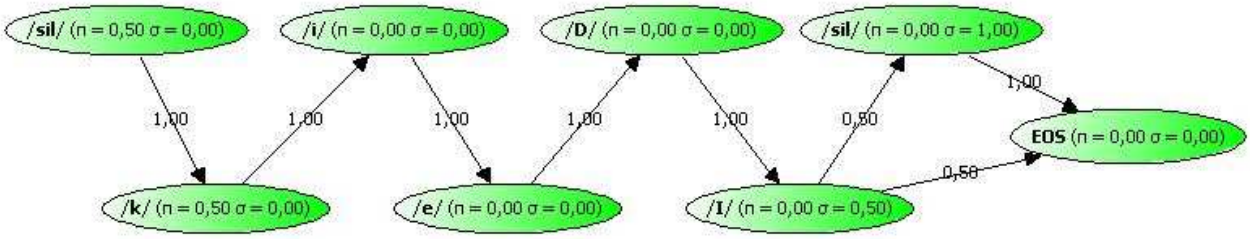
Reguły generowania HMM fonemów mogą być różne. Dobierając wartości **Pi**, **Sigma** oraz **A** możemy dopuszczać różne warianty wypowiedzi. W pracy zostały zastosowane następujące zasady:

[3] Reguły te nie muszą być spełnione, jeśli chcemy modelować więcej niż jeden wariant zdania w jednym modelu.

- Fonem ciszy zostaje dodany na początku i końcu słowa,
- Pi(.) = 0.5[4] dla fonemu ciszy na początku słowa,
- Pi(.) = 0.5 dla pierwszego fonemu w słowie (fonem ciszy nie musi wystąpić),
- Pi(.) = 0.00 dla pozostałych fonemów,
- Sigma(.) = 0.5 dla ostatniego fonemu w słowie (fonem ciszy nie musi wystąpić),
- Sigma(.) = 1 dla fonemu ciszy na końcu słowa,
- Sigma(.) = 0 dla pozostałych fonemów,
- **A**($fonem_N$,cisza) = 0.5 przejście z ostatniego fonemu słowa do ciszy,
- **A**($fonem_i$, $fonem_{i+1}$) = 1 przejścia pomiędzy kolejnymi fonemami. W p. p. 0,
- **B**($fonem_i$, $stan_i$) = 1 w przypadku emisji fonemu w odpowiadającym mu stanie. W p. p. 0.

Przykład:

Słowo: *kiedy* .
Zapis fonetyczny: /sil/, /k/, /i/, /e/, /D/, /I/, /sil/
HMM fonemów na rysunku 3.4:



Rysunek 3.4 – HMM fonemów

Poziom pod-fonemów

Ostatnim etapem generowania fonetycznego HMM jest rozbicie wszystkich fonemów w HMM fonemów na lewo-prawe modele pod-fonemowe. Tutaj w zależności od typu fonemu wyróżniamy kilka sposobów podziału na pod-fonemy:

Podział na 3 pod-fonemy

W ten sposób dzielimy na przykład normalne samogłoski. Występuje tu lewy i prawy kontekst fonemu oraz część centralna.

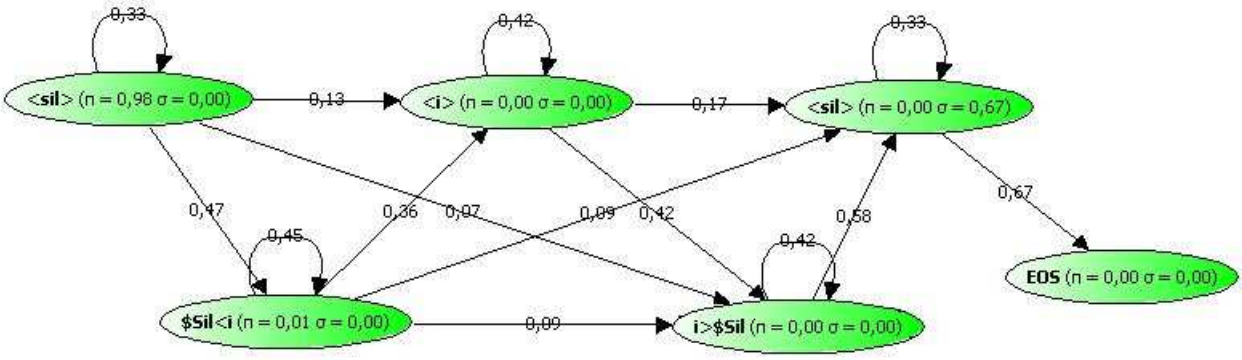
Przykład:

Dla sekwencji fonemów: /sil/ /i/ /sil/
Fonem /i/ zostanie rozbity na trzy części:

1. &Sil<i - fonem /i/ w kontekście lewostronnym ciszy
2. <i> - centralny fonem /i/
3. i>&Sil – fonem /i/ w kontekście prawostronnym ciszy

HMM wygenerowany w programie testowym znajduje się na rysunku 3.5.

[4] Wartości nie muszą być precyzyjnie dobrane, gdyż będą poddawane procesowi uczenia.

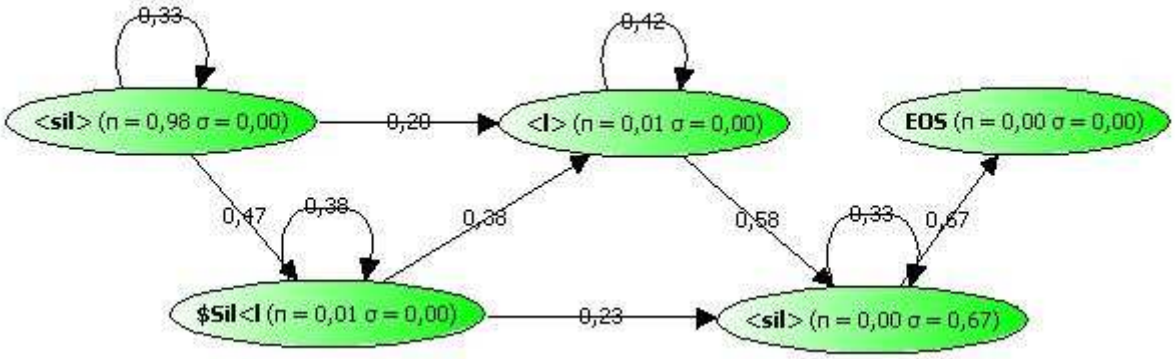


Rysunek 3.5 – Podział na 3 pod-fonemy

Podział na 2 pod-fonemy (lewo-zależny)
W tym przypadku fonem rozbijamy na dwa pod-fonemy. Jeden lewo-zależny i jeden centralny.

Przykład:

Fonemy: /sil/, /l/, /sil/
Wynik: HMM na rysunku 3.6.

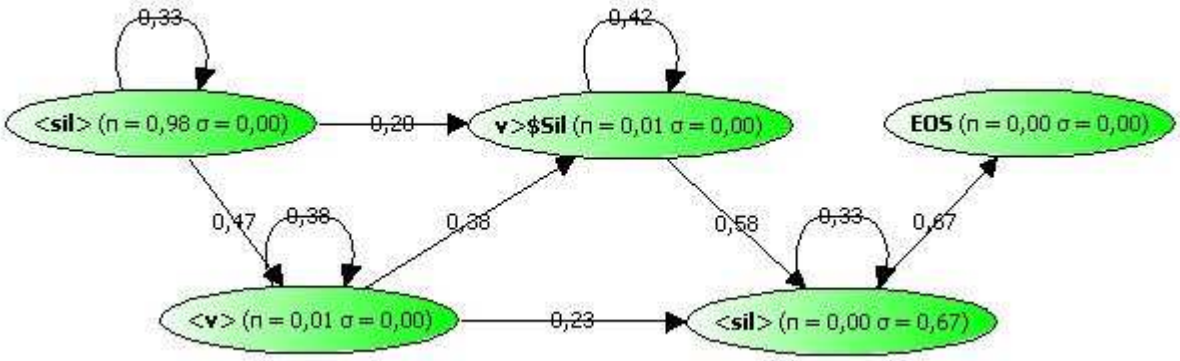


Rysunek 3.6 – Podział na 2 pod-fonemy (lewo zależny)

Podział na 2 pod-fonemy (prawo-zależny)
W tym przypadku fonem rozbijamy na dwa pod-fonemy. Jeden prawo-zależny i jeden centralny.

Przykład:

Fonemy: /sil/, /v/, /sil/
Wynik: HMM na rysunku 3.7.



Rysunek 3.7 – Podział na 2 pod-fonemy (prawo zależny)

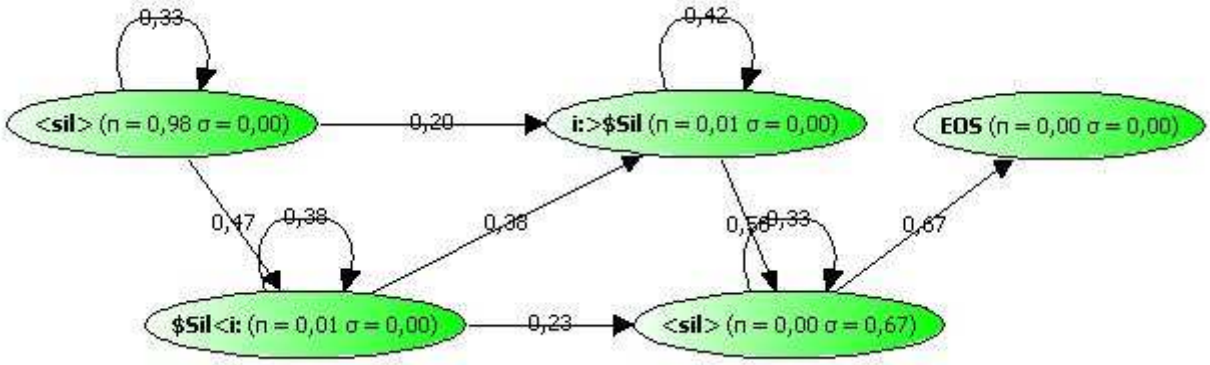
Podział na 2 pod-fonemy (tylko lewo i prawo zależny)
W tym przypadku fonem rozbijamy na dwa pod-fonemy. Jeden lewo-zależny i jeden

prawo-zależny. Nie występuje tu pod-fonem centralny. W taki sposób rozbija się między innymi skrócone samogłoski w języku angielskim.

Przykład:

Fonemy: /sil/, /i:/, /sil/

Wynik: HMM na rysunku 3.8.



Rysunek 3.8 – Podział na 2 pod-fonemy (lewo-prawo zależny)

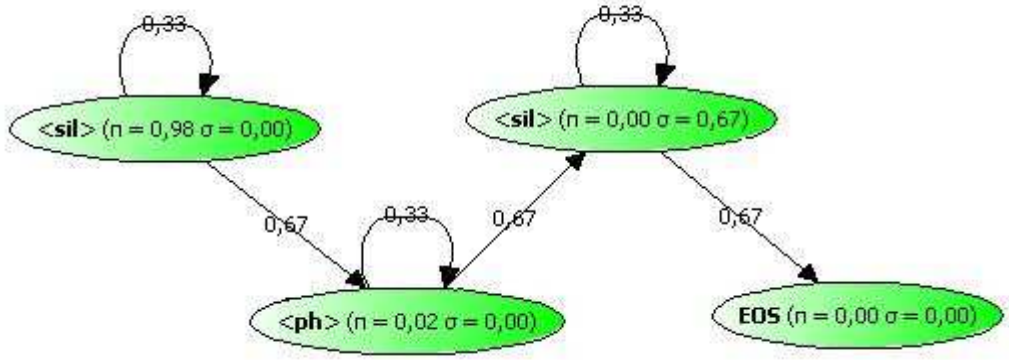
Podział na jeden pod-fonem (tylko centralny)

W tym przypadku fonem zamieniamy jedynie na pod-fonem centralny.

Przykład:

Fonemy: /sil/, /ph/, /sil/

Wynik: HMM na rysunku 3.9.



Rysunek 3.9 – Podział na 1 pod-fonem

Zwarcia krtaniowe (lewy pod-fonem)

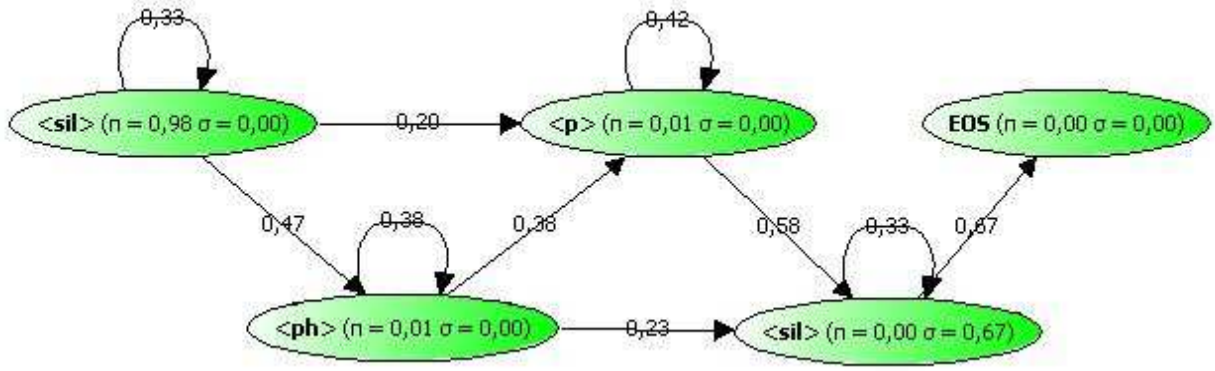
W tym przypadku lewo-zależny pod-fonem nie opisuje kontekstu, tylko typ zwarcia krtaniowego[5].

Przykład:

Fonemy: /sil/, /p/, /sil/

Wynik: HMM na rysunku 3.10.

[5] Szczególne traktowanie zwarć krtaniowych nie jest tu konieczne. Powinno przyczyniać się do przyspieszenia procesu nauki, jednak zwiększa ryzyko błędów w rozpoznawaniu.



Rysunek 3.10 – Podział na 2 pod-fonemy ze zwarciem krtaniowym

3.2 HIERARCHICZNY UKRYTY MODEL MARKOWA

Ostateczny pod-fonemowy HMM zdania uzyskamy łącząc modele pojedynczych fonemów w słowa oraz modele słów w zdania. Zbudujemy w ten sposób jeden hierarchiczny Ukryty Model Markowa.

Algorytm łączący modele fonemów w słowa wygląda następująco:

1. Utwórz pusty HMM (oznaczymy go NewHMM),
2. Dla każdego fonemu w HMM słowa (oznaczymy go SuperHMM):
   a. Utwórz pod-fonemowy HMM,
   b. Dodaj wszystkie stany pod-fonemowego HMM do NewHMM[6],
   c. Przypisz wszystkie stany pod-fonemowego HMM do odpowiadającego im stanu z SuperHMM jako podstany.
3. Dla wszystkich stanów s_i, s_j w SuperHMM:
   a. Dla wszystkich podstanów $s_{i,k}$, $s_{j,1}$ w NewHMM[7]:
      i. Ustaw przejście z $s_{i,k}$ do $s_{j,1}$: $\mathbf{A}(s_{i,k}, s_{j,1}) = \mathbf{A}(s_i, s_j) * \mathbf{Sigma}(s_{i,k}) * \mathbf{Pi}(s_{j,1})$
4. Dla wszystkich stanów s_i w SuperHMM:
   a. Dla wszystkich podstanów $s_{i,k}$ w NewHMM:
      i. $\mathbf{Pi}(s_{i,k}) = \mathbf{Pi}(s_i) * \mathbf{Pi}(s_{i,k})$
      ii. $\mathbf{Sigma}(s_{i,k}) = \mathbf{Sigma}(s_i) * \mathbf{Sigma}(s_{i,k})$

Zatem algorytm zakłada kolejno:
1. Utworzenie nowego HMM,
2. Wypełnienie nowego HMM stanami,
3. Określenie macierzy A dla nowego HMM,
4. Określenie macierzy **Pi** i **Sigma** dla nowego HMM.

Symbole emitowane w stanach określamy na podstawie kontekstu, w jakim znajduje się nadrzędny fonem. Jeśli rozbijamy fonem */o/*, do którego istnieje przejście z fonemu */sil/*, to lewy pod-stan będzie emitować symbol *sil<o* z prawdopodobieństwem 1. Jeśli istnieje więcej przejść do stanu reprezentującego */o/*, to w lewym pod-stanie będziemy mogli generować kilka odpowiadającym im symboli. Ich prawdopodobieństwa mogą być sobie równe, ponieważ i tak będą podlegać procesowi uczenia.

[6] Prawdopodobieństwa Pi, Sigma i A będziemy ustawiać później.
[7] $s_{i,k}$ oznacza k-ty podstany i-tego stanu w SuperHMM

Analogiczny algorytm wykorzystujemy do łączenia słów w zdania. Rolę fonemów pełnią wtedy słowa.

Implementacja

```
private Branch convertBranch(Branch branch) {
        log.info("Converting branch to sub-phonems: " + branch.getName());
        Branch hierarchicalSubphonemBranch = new Branch();
        hierarchicalSubphonemBranch.setName(branch.getName());

        final EndState endSubState = new EndState();
        for (State state : branch.getStates()) {
            if(state.isEndOfSentence()) {
                endSubState.setSuperState(state);
                break;
            }
        }
        for (State state : branch.getStates()) {
            createSubStates(state, hierarchicalSubphonemBranch, endSubState);
        }
        for (State state : branch.getStates()) {
            for (Entry<State, Double> entry : state.getArchMap().entrySet()) {
                final State nextState = entry.getKey();
                final Double probability = entry.getValue();
                connectStates(state, nextState, probability);
            }
        }
        for (State state : branch.getStates()) {
            setSubStatePiProbability(state);
            setSubStateSigmaProbability(state, endSubState);
        }

        hierarchicalSubphonemBranch.normalize();
        return hierarchicalSubphonemBranch;
    }
…
```

3.3 Uczenie parametrów modelu

Algorytm Bauma-Welcha wykorzystuje się w procesie uczenia HMM. Sam proces uczenia polega na takim zmodyfikowaniu macierzy **Pi, Sigma, A** i **B** by zmaksymalizować prawdopodobieństwo wygenerowania zbioru sekwencji uczących. Struktura modelu (ilość stanów, przejścia między stanami i wyprowadzane w nich symbole) jest w tym przypadku z góry narzucona.

Danymi wejściowymi algorytmu Bauma-Welcha jest HMM z początkowymi wartościami w macierzach **Pi, Sigma, A** i **B** oraz sekwencja obserwacji **O**. Warunkiem dodatkowym jest by sekwencja ucząca mogła być wygenerowana przez model z niezerowym prawdopodobieństwem.

Wyjściem algorytmu jest zmodyfikowany HMM, zmaksymalizowany pod kątem prawdopodobieństwa generowania sekwencji uczącej.

Algorytm składa się z kilku etapów. Pierwszym jest obliczenie wartości dwóch funkcji nazwanych tu $\alpha(t,\ k)$ i $\beta(t,\ k)$. Określają one prawdopodobieństwo wygenerowania t początkowych lub T-t końcowych symboli sekwencji przez ścieżkę przechodzącą przez stan k w chwili t. Następnie obliczana jest wartość oczekiwana liczby odwiedzin stanu i liczby przejść między stanami podczas generowania sekwencji. Na końcu ponawiana jest estymacja macierzy **A, B, Pi** i **Sigma**.

Prawdopodobieństwo wygenerowania sekwencji przez HMM

W pierwszym kroku zajmę się prawdopodobieństwem wygenerowania sekwencji obserwacji **O** przez HMM. Prawdopodobieństwo to jest sumą prawdopodobieństw wszystkich ścieżek, które generują sekwencję **O**. Możemy to zapisać w postaci:

$$P(O) = \sum_{Q} P(O, Q)$$

gdzie $P(O)$ – prawdopodobieństwo generowania sekwencji obserwacji O przez HMM, Q – dowolna ścieżka, a $P(O,\ Q)$ – prawdopodobieństwo generowania O przez ścieżkę Q. Aby obliczyć $P(O)$ możemy zastosować algorytm prefiksowy lub sufiksowy. Wartości wyznaczane przez te algorytmy będą potrzebne na dalszym etapie algorytmu B-W.

Algorytm prefiksowy

Zdefiniujmy sobie funkcję $\alpha(t,\ k)$, która określać będzie prawdopodobieństwo wygenerowania t początkowych symboli sekwencji przez dowolną ścieżkę kończącą się w stanie k. Możemy to zapisać równaniem:

$$\bigvee_{t=1..T} \bigvee_{k=1..N} \alpha(t,k) = \sum_{Q \wedge Q(t)=k} P(O[1..t], Q)$$

Dla $t = 0$, 1 zakładamy, że funkcja α przyjmuje wartości:

$$\bigvee_{k=1..N} \alpha(0,k) = 1$$

$$\bigvee_{k=1..N} \alpha(1,k) = B(O_1, s_k) * Pi(s_k)$$

Dla $t = 2, \ldots, |\mathbf{O}|$ funkcja α przyjmuje wartości:

$$\bigvee_{t=2..T} \bigvee_{k=1..N} \alpha(t,k) = B(O_t, s_k) * \sum_{l=1..N} \{\alpha(t-1, l) * A(l,k)\}$$

Gdy wyznaczymy wartość funkcji α dla $t = |\mathbf{O}| = T$ możemy wyznaczyć prawdopodobieństwo wygenerowania sekwencji obserwacji O:

$$P(O) = \sum_{k=1..N} \alpha(T,k)$$

Algorytm sufiksowy

Zdefiniujmy sobie funkcję $\beta(t,\ k)$, która określać będzie prawdopodobieństwo wygenerowania T-t końcowych symboli sekwencji przez dowolną ścieżkę rozpoczynającą się w stanie k. Możemy to zapisać równaniem:

$$\bigvee_{t=1..T} \bigvee_{k=1..N} \beta(t,k) = \sum_{Q \wedge Q(t)=k} P(O[t..T], Q)$$

Dla $t = T$ funkcja β przyjmuje wartość 1 ponieważ jest to prawdopodobieństwo wygenerowania

pustej sekwencji przy dowolnym stanie początkowym:

$$\bigvee_{k=1..N} \beta(T,k) = 1$$

Zachodzi również zależność (wyznaczenie prawdopodobieństwa sekwencji obserwacji):

$$P(O) = \sum_{k=1..N} \beta(1,k) * Pi(s_k)$$

Dla $t = T$-1, T-2, …, 1 funkcja β przyjmuje wartości:

$$\bigvee_{t=T-1..1} \bigvee_{k=1..N} \beta(t,k) = \sum_{l=1..N} \{B(O_{t+1}, s_l) * \beta(t+1,l) * A(k,l)\}$$

Estymacja

Prawdopodobieństwo wygenerowania przez ukryty model Markowa sekwencji obserwacji *O* możemy wyznaczyć ze wzoru:

$$\bigvee_{t=1..T} P(O) = \sum_{k=1..N} \alpha(t,k) * \beta(t,k)$$

Zachodzi również zależność:

$$P(O) = \sum_{k=1..N} \alpha(T,k) = \sum_{k=1..N} \beta(1,k) * Pi(k)$$

Zauważmy też, że w danej chwili *t* prawdopodobieństwo przejścia ze stanu *i* do stanu *j* wynosi:

$$\bigvee_{t=1..T} \bigvee_{i=1..N} \bigvee_{j=1..N} \xi(t,i,j) = \frac{\alpha(t,i) * A(i,j) * \beta(t+1) * B(t+1,j)}{P(O)}$$

Zatem, aby obliczyć wartość oczekiwaną liczby przejść ze stanu *i* do stanu *j* w dowolnej chwili czasu dla sekwencji **O** należy zsumować ξ po wszystkich chwilach czasowych:

$$\bigvee_{i=1..N} \bigvee_{j=1..N} \gamma(i,j) = \sum_{t=1..T} \xi(t,i,j)$$

Sumując również otrzymaną funkcję *γ(i, j)* po wszystkich stanach modelu *j* otrzymujemy wartość oczekiwaną ilości odwiedzin stanu *i* podczas generowania sekwencji obserwacji O :

$$\bigvee_{i=1..N} \gamma(i) = \sum_{j=1..N} \sum_{t=1..T} \xi(t,i,j)$$

Możemy również wyznaczyć wartość oczekiwaną ilości odwiedzin stanu *i* w chwili *t* .

$$\bigvee_{t=1..T} \bigvee_{i=1..N} \gamma(t,i) = \sum_{j=1..N} \xi(t,i,j)$$

Przyjmując, że Ukryty Model Markowa może emitować *M* różnych klas symboli Ω_m, zdefiniujmy sobie wartość oczekiwaną ilości emisji symbolu klasy Ω_m w stanie *i* jako funkcję:

$$\bigvee_{m=1..M} \bigvee_{i=1..N} \gamma(i,\Omega_m) = \sum_{t=1..T} \gamma(t,i) * \chi(O(t) \in \Omega_m)$$

Maksymalizacja

Celem algorytmu Bauma-Welcha jest zmaksymalizowanie prawdopodobieństwa wygenerowania przez HMM zadanego zbioru sekwencji uczących. Załóżmy, że mamy *K* sekwencji obserwacji. Zdefiniujmy funkcję określającą prawdopodobieństwo wygenerowania zbioru sekwencji uczących przez Ukryty Model Markowa:

$$P(O_1 \wedge O_2 \wedge \cdots \wedge O_K) = P(O_1) * P(O_2) * \cdots * P(O_K)$$

W praktyce, aby zlikwidować konieczność mnożenia wielu wartości bliskich zeru, iloczyn prawdopodobieństw zamieniany jest na sumę zlogarytmowanych prawdopodobieństw. Suma taka przyjmuje wartości mniejsze równe *0* i zwiększenie jej wartości odpowiada zwiększeniu prawdopodobieństwa wygenerowania zbioru sekwencji uczących. Nazwijmy tę sumę funkcją *Score* i zdefiniujmy jako:

$$Score(O_1 \wedge O_2 \wedge \ldots \wedge O_K) = \log P(O_1) + \log P(O_2) + \cdots + \log P(O_K)$$

Algorytm Bauma-Welcha ma maksymalizować wartość funkcji *Score*.

Jeśli zsumujemy wartości oczekiwane ilości przejść ze stanu *i* do stanu *j* w *K* sekwencjach uczących otrzymamy:

$$\bigvee_{i=1..N} \bigvee_{j=1..N} P(i,j) = \sum_{k=1..K} \gamma_k(i,j)$$

Jest to wartość oczekiwana ilości przejść ze stanu *i* do stanu *j* we wszystkich sekwencjach uczących. $\gamma_k(i,j)$ oznacza tu wartość funkcji $\gamma(i,j)$ wyznaczonej dla *k*-tej sekwencji uczącej.

Wartość oczekiwaną ilości odwiedzin stanu *i* we wszystkich sekwencjach uczących możemy zapisać jako:

$$\bigvee_{i=1..N} P(i) = \sum_{k=1..K} \sum_{j=1..N} \gamma_k(i,j) = \sum_{k=1..K} \gamma_k(i) = \sum_{j=1..N} P(i,j)$$

Zatem po wyznaczeniu wartości $P(i, j)$ i $P(i)$ możemy estymować nową wartość macierzy przejść **A** ukrytego modelu Markowa:

$$\bigvee_{i=1..N} \bigvee_{j=1..N} A(i,j) = \frac{P(i,j)}{P(i)}$$

Analogicznie możemy zsumować dla wszystkich sekwencji uczących wartości oczekiwane ilości emisji symbolu klasy Ω_m w stanie *i* :

$$\bigvee_{i=1..N} \bigvee_{m=1..M} E(i,\Omega_m) = \sum_{k=1..K} \gamma_k(i,\Omega_m),$$

oraz wartość oczekiwaną wszystkich emisji dowolnego symbolu w stanie *i* dla wszystkich sekwencji uczących:

$$\bigvee_{i=1..N} E(i) = \sum_{k=1..K} E(i,\Omega_m)$$

Możemy w ten sposób estymować nową wartość macierzy *B* prawdopodobieństw emisji symboli w stanach:

$$\bigvee_{i=1..N}\bigvee_{m=1..M} B(i,m) = \frac{E(i,\Omega_m)}{E(i)}$$

Zauważmy, że możemy również znaleźć wartość oczekiwaną rozpoczęcia sekwencji obserwacji w stanie *i*:

$$\bigvee_{i=1..N} D(i) = \sum_{k=1..K} \gamma_k(1,i)$$

Gdzie $\gamma_k(1,i)$ jest wartością funkcji $\gamma(t,i)$ dla k-tej sekwencji obserwacji.

Suma *D*(*i*) dla wszystkich stanów jest równa *K*, ponieważ prawdopodobieństwo rozpoczęcia zdania w dowolnym stanie wynosi *1*.

Możemy w ten sposób estymować nową wartość macierzy **Pi**, określającej prawdopodobieństwo rozpoczęcia sekwencji obserwacji w stanach:

$$\bigvee_{i=1..N} Pi(i) = \frac{D(i)}{K}$$

Zakładając, że w chwili *T* układ znajduje się w stanie końcowym s_{stop} możemy określić wartości macierzy **Sigma** jako:

$$\bigvee_{i=1..N} Sigma(i) = \frac{\Sigma_{k=1..K}\gamma_k(T-1,i)}{K}$$

W ten sposób w algorytmie Bauma-Welcha wyznaczamy nowe wartości macierzy **A, B, Pi** i **Sigma**. Można udowodnić, że wartość funkcji *Score* dla HMM z nowymi wartościami macierzy jest większa lub równa jej wartości dla pierwotnego modelu.

Algorytm Bauma-Welcha można stosować wielokrotnie, za każdym razem operując na wyznaczonym poprzednim razem HMM. Dzięki temu dochodzimy do maksimum lokalnego/globalnego funkcji *Score* w przestrzeni parametrów modelu. Dla różnych wartości początkowych parametrów możemy zbiegać w kolejnych iteracjach do różnych maksimów funkcji *Score*.

Implementacja

Algorytmy prefiksowy i sufiksowy:

```
    public double[][][] calulateForwardBackward(List<Observation> observations) {
…}
```

Estymacja i maksymalizacja:

```
    void baumWelchTraining(List<Observation> observations) {
        …
    }
```

Uśrednianie modeli

W praktyce nie musimy wyznaczać wartości macierzy **A, B, Pi** i **Sigma** dla wszystkich dostępnych sekwencji uczących. Możemy określić wartości tych macierzy dla jednej sekwencji uczącej i uśrednić ich wartości z poprzednią wersją modelu.

Uśrednianie można zrealizować przechowując w modelu licznik cykli uczenia. Jeśli model był wytrenowany za pomocą *k* sekwencji uczących, to dla *k*+1 sekwencji wyznaczamy model pomocniczy maksymalizujący prawdopodobieństwo generowania tej sekwencji. Następnie korygujemy wartości kolejnych macierzy oryginalnego modelu za pomocą średniej ważonej:

$$\text{nowa wartość} = \frac{\text{stara wartość} * k + \text{wartość dla nowej sekwencji}}{k+1}$$

Wyznaczanie średniej ważonej aktualnej kopii HMM z wersją zmaksymalizowaną pod kątem kolejnej sekwencji obserwacji:

```
void merge(Branch learnBranch) {
    for(State myState : states) {
        State yourState = learnBranch.getState(myState.getId());
        double myPi = myState.getPi();
        double yourPi = yourState.getPi();
        myState.setPi((myPi * mergeCounter + yourPi)/(mergeCounter + 1.0));
        for(State myStateTo : states) {
            State yourStateTo = learnBranch.getState(myStateTo.getId());
            double myA = myState.getA(myStateTo);
            double yourA = yourState.getA(yourStateTo);
            myState.setA(myStateTo, (myA * mergeCounter + yourA) /
                                        (mergeCounter + 1.0));
        }
        for(HmmSymbol mySymbol : myState.getSymbols()) {
            double myB = myState.getB(mySymbol);
            double yourB = yourState.getB(mySymbol);
            myState.setB(mySymbol, (myB * mergeCounter + yourB)/
                                        (mergeCounter + 1.0));
        }

    }
    mergeCounter += 1;
}
```

Wyznaczanie prawdopodobieństwa emisji symbolu

Sekwencja obserwacji rejestrowana podczas rozpoznawania mowy składa się z klas fonetycznych. Natomiast symbolami emitowanymi przez fonetyczny HMM są pod-fonemy (zobacz punkt 5.1). Z tego względu musimy stosować dwie macierze prawdopodobieństwa emisji symbolu: $\boldsymbol{B}^{klasa}$ oraz $\boldsymbol{B}^{fonem}$. Prawdopodobieństwa emisji pod-fonemu $\boldsymbol{B}^{fonem}$ mają znaczenie wyłącznie w kontekście rozpatrywanego HMM. Wzory wyznaczające $\boldsymbol{B}^{fonem}$ można łatwo wyznaczyć:

Niech $\gamma(i, \Omega_m)$ będzie wartością oczekiwaną emisji symbolu Ω_m w stanie s_i (wyznaczamy ją podczas treningu Bauma-Welcha). Wtedy wartość oczekiwana wyemitowania fonemu f_m w stanie s_i jest opisane wzorem:

$$\bigvee_{i=1..N} \bigvee_{m=1..M} C^{fonem}(i,m) = \sum_{t=0}^{T} \gamma(i, \Omega_t) * B^{klasa}(m, \Omega_t)$$

Prawdopodobieństwo wyemitowania f_m w stanie s_i otrzymujemy po normalizacji:

$$\bigvee_{i=1..N} \bigvee_{m=1..M} B^{fonem}(i,m) = \frac{C^{fonem}(i,m)}{\sum_{m=0}^{M} C^{fonem}(i,m)}$$

W ten sposób ustalamy wartości macierzy B^{fonem}, którą zapamiętujemy w modelu HMM.

Wartości macierzy $\boldsymbol{B}^{klasa}$ nie powinny być zależne od modelu HMM. Jej inicjalne wartości możemy ustalić przyjmując, że prawdopodobieństwo wyemitowania klasy fonetycznej jest na początku takie samo dla wszystkich pod-fonemów (zakładamy istnienie N typów pod-fonemów):

$$\bigvee_{k=1..N} \bigvee_{m=1..M} B^{klasa}(k, \Omega_m) = \frac{1}{N}$$

W trakcie uczenia wykorzystujemy wartość oczekiwaną emisji klasy fonetycznej w stanie $\gamma(i, \Omega_t)$ do przyporządkowania klas fonetycznych do pod-fonemów. Sumujemy wartości oczekiwane emisji klasy fonetycznej przez pod-fonem i przypisujemy je do pod-fonemów. Sumę tę oznaczymy jako *K*:

$$\bigvee_{k=1..K} \bigvee_{m=1..M} K(fon_k, klasa_m) = \sum_{t=1..T} \sum_{i=1..N} B^{fonem}(stan_i, fon_k) * \gamma(stan_i, obs_t)$$

Następnie wyznaczamy wartości $\boldsymbol{B}^{klasa}$ według wzoru:

$$\bigvee_{k=1..K} \bigvee_{m=1..M} B^{klasa}(fon_k, klasa_m) = \frac{1 + K(fon_k, klasa_m)}{N + \sum_{s=1..M} K(fon_k, klasa_s)}$$

Implementacja

Funkcja z klasy **SubPhonem**. Każdy obiekt klasy **SubPhonem** przechowuje mapę *frameToHitsMap*. Mapa ta zawiera sumę powstałych w wyniku uczenia wartości oczekiwanych emisji klasy fonetycznej przez pod-fonem. Zawiera też licznik *allHits*, który jest sumą wszystkich wartości z mapy *frameToHitsMap*. Służy on do normalizacji wyników.

```
public double probabilityOfMatch(FrameObservation observation){
```

```
        double observationHits = frameToHitsMap.get(observation.getName());
        return (1 + observationHits) / (subPhonems.size() + allHits);
    }
```

Funkcja z klasy **SubPhonem** zapisująca wyniki treningu Bauma-Welcha:

```
    public void updateObservation(Observation observation, double prob) {
        allHits += prob;
        Double hits = frameToHitsMap.get(observation.getName());
        frameToHitsMap.put(observation.getName(),hits + prob);
    }
```

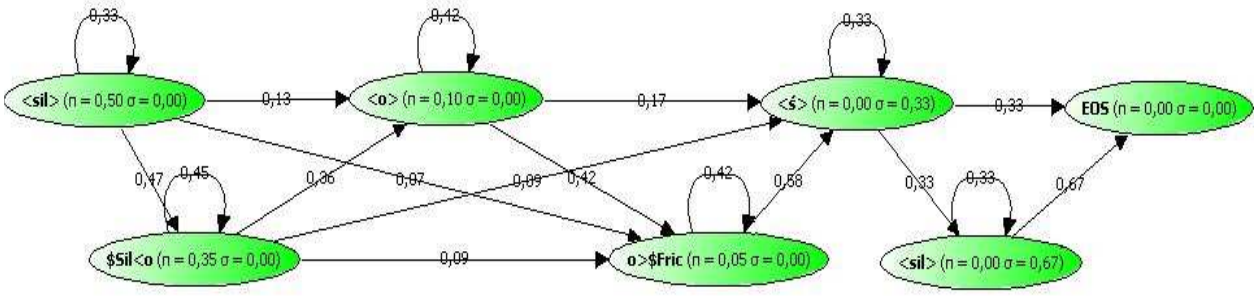
Część funkcji realizującej algorytm Bauma-Welcha, która aktualizuje B^{klasa}:

```
int i=0;
for(State stateI : states) {
  Collection<HmmSymbol> symbols = stateI.getSymbols();
  for(SubPhonem sym : symbols) {
     for(int t = 0; t < T; t++) {
        sym.updateObservation(observations.get(t), gammaTI[t][i]*stateI.getB(sym));
     }
   }
   i++;
}
```

Zaprezentowany sposób wyznaczania związku pomiędzy klasami fonetycznymi i pod-fonemami działa w sprzężeniu zwrotnym z algorytmem Bauma-Welcha. W testach system już po kilku sekwencjach uczących znajdował wstępne przyporządkowanie, które wraz z kolejnymi sekwencjami stawało się coraz dokładniejsze.

Przykład:
Dane jest zdanie „oś”.

Krok 0:
Wygenerowany automatycznie HMM słowa posiada narzucone z góry prawdopodobieństwa przejść i wyjść (rysunek 3.11).



Rysunek 3.11 – Model „oś” – wygenerowany

Macierz opisująca przyporządkowanie klas fonetycznych do pod-fonemów jest pusta.

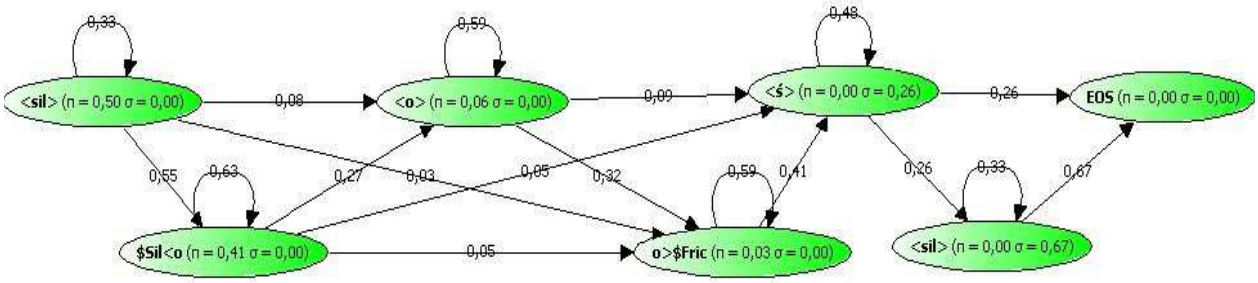
Krok 1:

Uruchamiamy procedurę uczącą:

1. Tworzymy kopię modelu słowa,

2. Maksymalizujemy kopię modelu pod kątem próbki uczącej,
3. Poprawiamy oryginalny model obliczając średnią ważoną z wartości oryginalnej i kopii (wagi w tym przypadku wynoszą 1 : 1),
4. Poprawiamy wyznaczone przyporządkowanie klas fonetycznych do pod-fonemów.

W wyniku otrzymujemy model przedstawiony na rysunku 3.12:



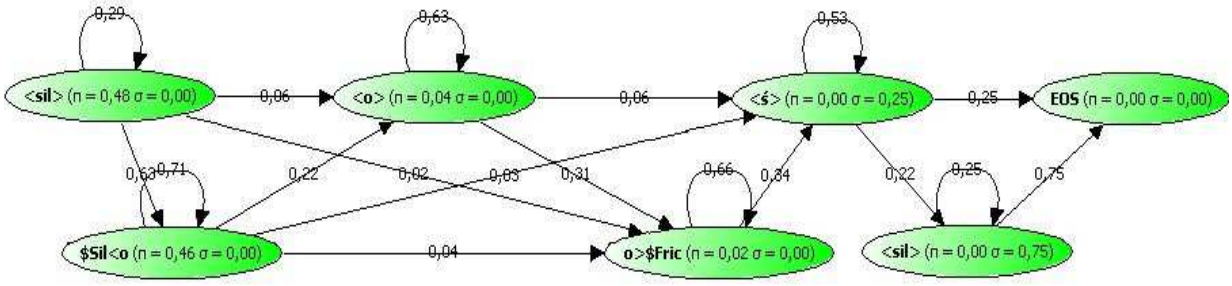
Rysunek 3.12 – Model „oś” – 1 cykl uczenia

Wstępne przyporządkowanie klas fonetycznych do pod-fonemu <o>:

o		MIDDLE	
	29	1.2296181670073614	(najlepiej dopasowana klasa)
	4	0.9674683551153661	
	46	0.43636851280246186	
	20	0.42690262455959843	
	39	0.4227623602895619	
	58	0.39460820989184436	
	33	0.0188244550264906	

Krok 2:

Uruchamiamy procedurę uczenia dla kolejnej próbki. Tym razem jednak obliczamy średnią ważoną z wagami 2 : 1 dla oryginalnego modelu. W wyniku otrzymujemy model przedstawiony na rysunku 3.13:



Rysunek 3.13 – Model „oś” – 2 cykl uczenia

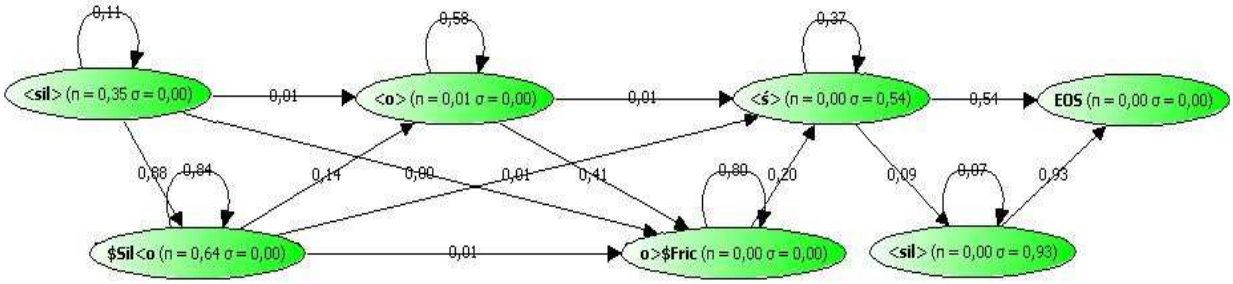
Model dopasowuje się coraz lepiej do sekwencji uczących. Przyporządkowanie klas fonetycznych do pod-fonemów również staje się wyraźniejsze:

o		MIDDLE
	29	1.6368642420657809
	4	1.3830834433199408
	20	0.9317465897761392
	46	0.930588313545302

39	0.8016299508914242
58	0.7895605642086205
6	0.5281316532039521
33	0.0188244550264906
45	0.005652338817262757
18	0.0011829907242758436

Krok N:

Powtarzamy cykl kolejno dla każdej próbki uczącej. Wagi w średniej ważonej stają się kolejno 3:1, 4:1, … N:1. Po 15 cyklach nauki model zdania „oś” wygląda jak na rysunku 3.14:



Rysunek 3.14 – Model „oś” – 15 cykl uczenia

Jak widać, struktura modelu pozostała bez zmian. Okazało się, że początkowy i końcowy fonem ciszy rzadko występuje. Najbardziej prawdopodobne okazało się pozostanie w stanie lub przejście do stanu następnego. Przyporządkowanie klas fonetycznych:

o		MIDDLE
	20	6.974903226756644
	43	6.015949026032915
	39	3.7701903350984125
	46	3.4939352686970326
	58	2.6147788925499134
	6	2.6047968717984293
	29	2.1916112334158413
	4	2.1791724274208226

Przyporządkowanie klas fonetycznych dla pod-fonemu <o> zmieniło się. Klasy, które wcześniej były na pierwszych miejscach przesunęły się niżej. Jest to naturalne w tym przypadku, ponieważ wstępne przyporządkowanie miało rozkład równomierny (każda klasa miała tą samą wagę). System dopiero w kolejnych fazach nauki znajduje właściwe przyporządkowanie. Proces uczenia powinien obejmować zróżnicowany zestaw zdań. Dzięki temu nie będzie się utrwalać wstępnie błędne przyporządkowanie klas fonetycznych.

4. Rozpoznawanie słów kluczowych

Głównym algorytmem stosowanym do rozpoznawaniu słów na podstawie sekwencji obserwacji jest **przeszukiwanie Viterbiego**. Znajduje on najlepszą ścieżkę w Ukrytym Modelu Markowa, która generuje wejściową sekwencję obserwacji. Znając najlepszą ścieżkę można obliczyć prawdopodobieństwo wygenerowania zadanej sekwencji obserwacji.

Szukając słów kluczowych w dłuższej sekwencji obserwacji musimy dokonać modyfikacji algorytmu przeszukiwanie Viterbiego. Zmiany te obejmują:

- Brak kary za pominięcie obserwacji poprzedzających słowo,
- Brak kary za pominięcie obserwacji następujących po słowie,
- Warunek przejścia przez cały model słowa,
- Brak kary za długość sekwencji,
- Ograniczona maksymalna ilość powtórzeń stanu.

W dalszej części rozdziału opisano algorytm przeszukiwanie Viterbiego w ogólnie znanej postaci. Następnie przedstawiono jak wykorzystać go do rozpoznawania słów kluczowych w wypowiedzi.

4.1 Algorytm „przeszukiwanie Viterbiego”

Przeszukiwanie Viterbiego służy odnajdywaniu najbardziej prawdopodobnej sekwencji stanów, które mogły wyemitować zaobserwowane słowo na wyjściu HMM. Na wejściu algorytmu mamy HMM opisany macierzami **A, B, Sigma**, oraz **Pi** i zaobserwowane słowo O (sekwencja wyprowadzanych przez model symboli). Na wyjściu otrzymujemy najbardziej prawdopodobną sekwencję stanów i prawdopodobieństwo tej sekwencji.

Algorytm Viterbiego jest odmianą programowania dynamicznego. Zdefiniujemy sobie funkcję $g(t, k)$, która określa prawdopodobieństwo optymalnej drogi kończącej się w stanie s_k i która generuje sekwencje obserwacji $\mathbf{O}[1..t]$.

Dla $t=1$ przypisujemy funkcji g wartości:

$$\bigvee_{k=1..N} g(1,k) = B(O_1, s_k) * Pi(s_k)$$

Dla kolejnych chwil czasowych $t = 2, \ldots, T$ obliczamy wartość funkcji g (jak również najlepsze ścieżki, które mogły wygenerować prefiks słowa o długości t) za pomocą równania rekurencyjnego:

$$\bigvee_{t=2..T} \bigvee_{k=1..N} g(t,k) = B(O_t, s_k) * \max_{l=1..N} \left(g(t-1, l) * A(l,k) \right),$$

gdzie O_t jest indeksem symbolu obserwacji $O(t)$, natomiast $B(O_t, s_k)$ jest prawdopodobieństwem wyemitowania symbolu $O(t)$ w stanie s_k [8]. $\mathbf{A}(l, k)$ jest prawdopodobieństwem przejścia ze stanu s_l do stanu s_k.

Korzystamy tu z cechy ukrytych modeli Markowa, którą jest niezależność prawdopodobieństwa przejścia między stanami od historii dojścia do stanu, z którego wychodzimy. Zatem rozpatrując prefiks zaobserwowanego słowa o długości t i znając najbardziej prawdopodobne ścieżki, które

[8] Dotyczy to ukrytych modeli Markowa z dyskretnym zbiorem symboli wyjściowych. Dla modeli, które mogą generować ciągły zbiór symboli wyjściowych należy zamienić elementy macierzy **B** przez wyniki funkcji rozkładu prawdopodobieństwa wygenerowania symbolu w stanie.

wyemitowały prefiks o długości t-1 i skończyły się w stanach s_1, s_2, …, s_N możemy znaleźć ścieżki o 1 dłuższe.

Po wyznaczeniu optymalnej ścieżki dla całej sekwencji wyjściowej, wyznaczana jest również najbardziej prawdopodobna sekwencja stanów dla obserwowanego słowa.

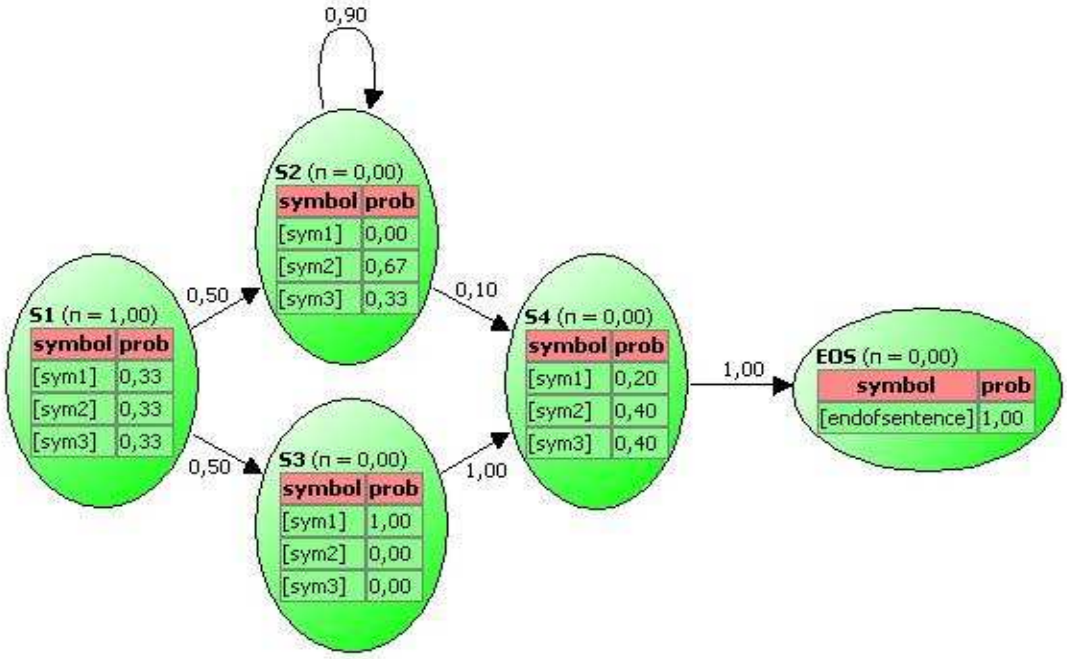
Implementacja

Poniżej znajduje się algorytmiczny zapis przeszukiwania Viterbiego:

// Parametry algorytmu Viterbiego
O // Obserwowana sekwencja symboli
$T = |O|$ // Długość obserwowanej sekwencji
$N = |\{s_1, s_2, \ldots, s_N\}|$ // Ilość stanów w HMM
$M = |\{sym_1, sym_2, \ldots, sym_M\}|$ // Ilość symboli alfabetu wyprowadzanych przez HMM
$\mathbf{Pi}_{Nx1}$ // Wektor prawdopodobieństw rozpoczęcia sekwencji w stanie
$\mathbf{A}_{NxN}$ // Macierz prawdopodobieństw przejść między stanami HMM
$\mathbf{B}_{NxM}$ // Macierz prawdopodobieństw wyprowadzenia symbolu w stanie

// Inicjalizacja stanów HMM
FOR j = 1, 2, …, N DO { // Dla wszystkich stanów
G(j, 0) = **B**(j, I(O(0))) · **Pi**(j) // Początkowa waga
ψ(j, 0) = 0. // Pusta ścieżka
}
FOR t = *1*, *2*, …, T DO { // Dla całej zaobserwowanej sekwencji
// Znajdź najlepszą krawędź
1) G(j, t) = **B**(j, I(O(t))) · { **max** $_{i\in\{1;\ldots;N\}}$ [G(t-1, i) · **A**(i, j)] }
// Zapamiętaj indeksy krawędzi prowadzącej maksimum G(t, j).
2) ψ(j, t) = **arg max** $_{i\in\{1;\ldots;k\}}$ [G(t-1, i) · **A**(i, j)] // Indeks najlepszego
// poprzedniego stanu
}
// Zakończenie
FOR j = 1, ..., N DO { // Dla wszystkich stanów HMM
G(T+1, j) = G(T, j) · **A**(i, #) // Prawdopodobieństwo zakończenia sekwencji w stanie j
}
// Określ najlepszy łuk
$G(\Omega \mid \boldsymbol{C})$ = **max** $_{j\in\{1;\ldots;k\}}$ G(j, T+1)
Q(T) = **arg max** $_{j\in\{1;\ldots;k\}}$ G(j, T+1)
// Przejdź wstecz po zapamiętanych krawędziach – optymalna ścieżka
FOR t = ***T-1,*** …, 1 DO {
Q(t) = ψ(Q(t+1), t + 1)
}
// Wynik – najlepsza ścieżka, prawdopodobieństwo wygenerowania sekwencji obserwacji
RETURN: Q, $G(\Omega \mid \boldsymbol{C})$

Przykład 1:
Załóżmy, że zaobserwowaliśmy sekwencję symboli: **O** = {sym1, sym1, sym2}. Ukryty model Markowa przedstawiono na rysunku 4.1



Rysunek 4.1 - Ukryty model Markowa dla przykładu 1.

Zastosujemy algorytm Viterbiego do znalezienia optymalnej ścieżki w ukrytym modelu Markowa, to znaczy takiej, która w najlepszy sposób mogła wyemitować zaobserwowaną sekwencję symboli (słowo):

// Parametry algorytmu Viterbiego
O = {sym1, sym1, sym2} // Obserwowana sekwencja symboli
T = |O| = 3 // Długość obserwowanej sekwencji
N = |{s_1, s_2, s_3, s_4}| = 4 // Ilość stanów w HMM
M = |{sym1, sym2, sym3}| = 3 // Ilość symboli alfabetu wyprowadzanych przez HMM

$$Pi = \begin{bmatrix} 1.00 \\ 0.00 \\ 0.00 \\ 0.00 \end{bmatrix}$$ //Prawdopodobieństwa rozpoczęcia sekwencji w stanie

$$A = \begin{bmatrix} 0.00 & 0.50 & 0.50 & 0.00 \\ 0.00 & 0.90 & 0.00 & 0.10 \\ 0.00 & 0.00 & 0.00 & 1.00 \\ 0.00 & 0.00 & 0.00 & 0.00 \end{bmatrix}$$ // Prawdopodobieństwa przejść między stanami HMM

$$B = \begin{bmatrix} 0.33 & 0.33 & 0.33 \\ 0.00 & 0.67 & 0.33 \\ 1.00 & 0.00 & 0.00 \\ 0.20 & 0.40 & 0.40 \end{bmatrix}$$ // Prawdopodobieństwa wyprowadzenia symbolu w stanie

// Inicjalizacja stanów HMM
FOR j = 1, 2, …, N DO { // Dla wszystkich stanów
G(j, 1) = B(j, I(O(1))) · **Pi**(j) // Początkowa waga
ψ (j, 1) = 0 // Pusta ścieżka
}

$$G = \begin{bmatrix} 0.33 & * & * & * \\ 0.00 & * & * & * \\ 0.00 & * & * & * \\ 0.00 & * & * & * \end{bmatrix} \quad \Psi = \begin{bmatrix} 0 & * & * \\ 0 & * & * \\ 0 & * & * \\ 0 & * & * \end{bmatrix}$$

FOR t = 2, …, T DO { // Dla całej zaobserwowanej sekwencji
// Znajdź najlepszą krawędź
1) G(j, t) = B(j, I(O(t))) · { **max** $_{i\in\{1;\ldots;N\}}$ [G(t-1, i) · A(i, j)] }
// Zapamiętaj indeksy krawędzi prowadzącej maksimum G(t, j).
2) ψ(j, t) = **arg max** $_{i\in\{1;\ldots;k\}}$ [G(t-1, i) · A(i, j)] // Indeks najlepszego stanu
}

$$G = \begin{bmatrix} 0.33 & 0.00 & 0.00 & * \\ 0.00 & 0.00 & 0.00 & * \\ 0.00 & 0.165 & 0.00 & * \\ 0.00 & 0.00 & 0.066 & * \end{bmatrix} \quad \Psi = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

// Zakończenie
FOR j = *1*, ..., N DO { // Dla wszystkich stanów HMM
G(j, T+1) = G(j, T) · A(i, EOS) // Prawdopodobieństwo zakończenia sekwencji
// w stanie j
}

$$G = \begin{bmatrix} 0.33 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.165 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.066 & 0.066 \end{bmatrix} \quad \Psi = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

// Określ najlepszy łuk
G (Ω | $\boldsymbol{C}$) = **max** $_{j\in\{1;\ldots;k\}}$ G(j, T+1)
Q(T) = **arg max** $_{j\in\{1;\ldots;k\}}$ G(j, T+1)

G(Ω | $\boldsymbol{C}$) = 0.066
Q(t=3) = 4

// Przejdź wstecz po zapamiętanych krawędziach – optymalna ścieżka
FOR t = *T-1*, …, 1 DO {
Q(t) = ψ(Q(t+1), t + 1)
}

// Wynik – najlepsza ścieżka, prawdopodobieństwo wygenerowania sekwencji obserwacji
RETURN: Q =(1, 3, 4}, G (Ω | $\boldsymbol{C}$) = 0.066

Przykład 2:

Najlepsza ścieżka wyznaczona przez algorytm Viterbiego[9] dla sekwencji klas fonetycznych słowa “zero”:

Klasa Fonetyczna	**Pod-fonem (rzeczywisty)**	**Stan (rozpoznawanie)**
3	<sil>	<sil>
5	<z>	<z>
6	z>$Front	z>$Front
6	z>$Front	z>$Front
6	z>$Front	z>$Front
6	z>$Front	z>$Front
6	z>$Front	z>$Front
8	$Fric<e	$Fric<e
8	$Fric<e	$Fric<e
7	<e>	<e>
7	<e>	<e>
13	e>$Other	e>$Other
75	$Front<r	$Front<r
74	<r>	<r>
74	<r>	<r>
74	<r>	<r>
74	<r>	<r>
74	<r>	<r>
74	<r>	<r>
74	<r>	<r>
74	<r>	<r>
74	<r>	<r>
55	$Other<o	$Other<o
34	<o>	<o>
56	o>$Sil	o>$Sil
3	<sil>	<sil>
3	<sil>	<sil>
3	<sil>	<sil>

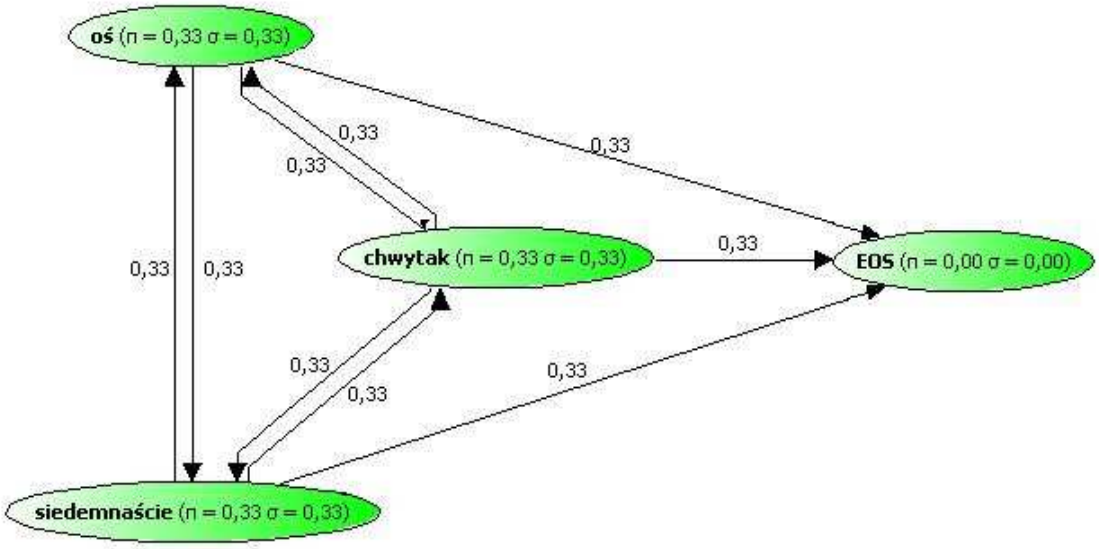
[9] W tabeli przedstawiono wynik symulacji.

4.2 KLASYFIKACJA SŁÓW

Klasyfikację słów kluczowych możemy podzielić na dwie kategorie. W pierwszej kategorii zakładamy, że wypowiedź składa się wyłącznie ze słów kluczowych. W drugiej takiego założenia nie robimy.

Klasyfikacja wypowiedzi złożonej ze słów kluczowych

Jeśli bierzemy pod uwagę tylko pierwszą kategorię, dobrym podejściem jest zbudowanie odpowiedniego HMM i zastosowanie klasycznego przeszukiwania Viterbiego.
Przykładowo dla słów kluczowych „oś", „chwytak" oraz „siedemnaście" możemy zbudować model jak na rysunku 4.2:



Rysunek 4.2 – HMM dla 3 słów kluczowych

Taki model będzie dawał dobre wyniki przy założeniu, że w wypowiedzi nie pojawią się słowa zupełnie nieznane.

Klasyfikacja słowa w dowolnej wypowiedzi
W procesie klasyfikacji słowa w środku zdania musimy ignorować część poprzedzającą słowo i następującą po nim. W wyniku mamy zwrócić dopasowanie części obserwacji do HMM słowa.

Ignorowanie początkowej i końcowej części sekwencji obserwacji możemy zapewnić rozbudowując HMM słowa o dwa stany:

- *AnyFonemBefore* – dowolny pod-fonem poprzedzający słowo
- *AnyFonemAfter* – dowolny pod-fonem następujący za słowem

Dodatkowo stan *AnyFonemBefore* musi spełniać warunku:

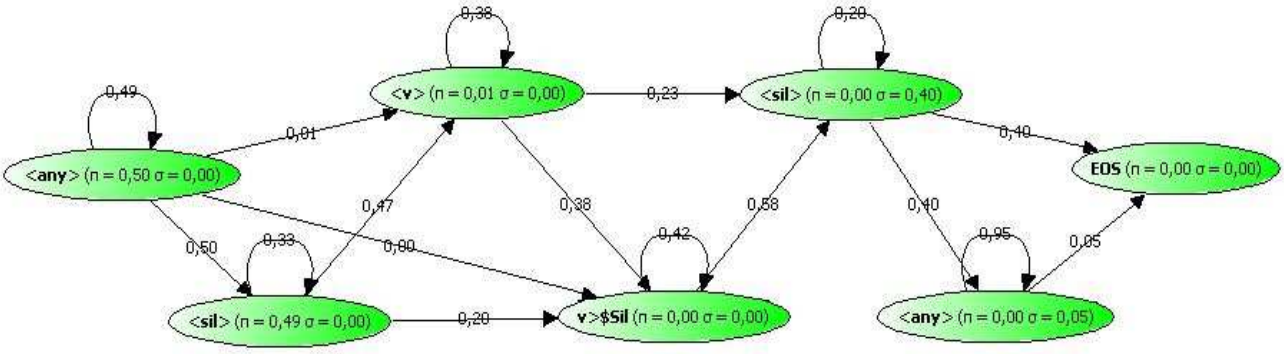
- Wartość **Pi()** dla tego stanu musi być niezerowa (sekwencja obserwacji może się rozpocząć od tego stanu),
- Wartość **Sigma()** dla tego stanu musi wynosić zero (nie możemy ominąć słowa i przejść od razu do stanu końcowego),
- Stan *AnyFonemBefore* musi akceptować dowolny ciąg symboli z pominięciem kary,

- Przejścia do stanów opisujących słowo są wykonywane z prawdopodobieństwem **Pi** tych stanów.

Analogicznie stan *AnyFonemAfter* musi spełniać warunku:

- Wartość **Pi()** dla tego stanu musi być zerowa,
- Wartość **Sigma()** dla tego stanu musi wynosić jeden,
- Stan *AnyFonemAfter* musi akceptować dowolny ciąg symboli z pominięciem kary,
- Przejścia od stanów opisujących słowo do *AnyFonemAfter* są wykonywane z prawdopodobieństwem **Sigma** tych stanów.

Przykład:
Model słowa „w” rozbudowany o dwa stany akceptujące początkową i końcową część sekwencji obserwacji został przedstawiony na rysunku 4.3:[10]



Rysunek 4.3 – HMM klasyfikujący słowo

Parametry stanów *AnyFonemBefore* i *AnyFonemAfter*
W zaprezentowanym mechanizmie istotne jest specjalne traktowanie w algorytmie Viterbiego stanów *AnyPhonemBefore* oraz *AnyPhonemAfter*.

Podczas szukania najlepszej ścieżki początkowa jakość wynosi 0. Dopóki ścieżka pozostaje w stanie *AnyPhonmeBefore* jakość się nie zmienia i ciągle wynosi 0. Jeśli ścieżka wychodzi ze stanu *AnyPhonemBefore*, traktujemy to jak rozpoczęcie sekwencji obserwacji słowa. Z tego względu prawdopodobieństwo takiego przejścia obliczamy na podstawie macierzy *Pi* oraz prawdopodobieństwa wygenerowania obserwacji.

Dojście do stanu *AnyPhonemAfter* następuje po przejściu przez cały model zdania. W tym stanie wybieramy więc najlepszą ścieżkę dla całej sekwencji obserwacji (szukamy maksimum). Po wybraniu najlepszej ścieżki akceptujemy resztę obserwacji z tą samą jakością.

Opisane modyfikacje algorytmu Viterbiego pozwalają znaleźć najlepszą ścieżkę pasującą do słowa jako część dłuższego zdania. Nie są jednak wystarczające by odnaleźć właściwą ścieżkę i by stwierdzić czy słowo występuje w zdaniu. Występuje tu problem skracania ścieżki do minimum, ponieważ nie istnieje warunek przejścia przez całą sekwencję obserwacji, a kara za pozostanie w którymś z dodatkowych stanów jest zerowa. Rozwiązanie tego problemu polega na takim obliczaniu jakości ścieżki, by dla poprawnej sekwencji klas fonetycznych jakość była zbliżona do zera. Wymaga to wyznaczenia średniej jakości oraz średniej długości poprawnych

[10] W uzupełnionym modelu nie powinniśmy wykonywać normalizacji, ponieważ stany *AnyPhonemBefore* i *AnyPhonemAfter* trzeba traktować szczególnie w algorytmie Viterbiego.

ścieżek w procesie uczenia. Gdy posiadamy te dwie wartości powinniśmy dla każdej obserwacji zwiększać jakość ścieżki przechodzącej przez stany modelu słowa o współczynnik:

$$NormalizationFactor = -\frac{Mean\ quality}{Mean\ length}$$

Kolejnym ważnym elementem rozpoznawania słów jest zapewnienie, by ilość powtórzeń stanu nie mogła rosnąć w nieskończoność. Proponowanym rozwiązaniem może być zwiększanie kary za pozostanie w stanie według wzoru:

$$Kara = ilość\ powtórzeń\ stanu * (1 - prawdopodobieństwo\ pozostania\ w\ stanie)$$

Rosnąca kara zależna od macierzy prawdopodobieństw przejść promuje wyszukiwanie ścieżek zgodnych z próbkami uczącymi wykorzystywanymi w procesie uczenia.

Decyzja określająca czy słowo występuje w zdaniu mogłaby brać pod uwagę dwa czynniki:

- Jakość znalezionej ścieżki,
- Długość znalezionej ścieżki.

Implementacja rozszerzonego algorytmu Viterbiego:

```
public Path viterbiWordProb(List<Observation> observations) {

…

}
```

4.3 TESTY WYSZUKIWANIA SŁOWA W ZDANIU

Przykład 1:
Szukamy słowa „oś” w wypowiedzi zdania „oś dwa dół start stop”. Najlepsza znaleziona ścieżka dla modelu słowa „oś” i powyższej obserwacji (zdania) wygląda następująco:

KOD	POD-FONEM	JAKOŚĆ ŚCIEŻKI	KOMENTARZ
0	<ANY>	0.0	CISZA PRZED SŁOWEM (JAKOŚĆ RÓWNA 0)
0	<ANY>	0.0	CISZA PRZED SŁOWEM (JAKOŚĆ RÓWNA 0)
0	<ANY>	0.0	CISZA PRZED SŁOWEM (JAKOŚĆ RÓWNA 0)
163	<ANY>	2.822363219756552	ROZPOCZĘCIE SŁOWA (NA PODSTAWIE MACIERZY PI)
166	$SIL<O	4.62921864264894	SŁOWO
166	$SIL<O	5.685027751104923	SŁOWO
99	$SIL<O	6.922744583296412	SŁOWO
206	$SIL<O	6.673610446663559	SŁOWO
96	<O>	7.37011636541062	SŁOWO
79	<O>	8.68648649747712	SŁOWO
141	O>$FRIC	10.54048667484746	SŁOWO
141	O>$FRIC	12.058957913900633	SŁOWO
152	O>$FRIC	13.349981491681856	SŁOWO
159	O>$FRIC	14.208488287623695	SŁOWO
137	O>$FRIC	14.550151833976047	SŁOWO
154	<Ś>	15.639141074692487	SŁOWO
154	<Ś>	15.866173440532966	SŁOWO
160	<Ś>	13.584543680898175	SŁOWO
0	<ANY>	13.584543680898175	CISZA ZA SŁOWEM (JAKOŚĆ SIĘ NIE ZMIENIA)
142	<ANY>	13.584543680898175	RESZTA ZDANIA (JAKOŚĆ SIĘ NIE ZMIENIA)
7	<ANY>	13.584543680898175	RESZTA ZDANIA (JAKOŚĆ SIĘ NIE ZMIENIA)
30	<ANY>	13.584543680898175	RESZTA ZDANIA (JAKOŚĆ SIĘ NIE ZMIENIA)
30	<ANY>	13.584543680898175	RESZTA ZDANIA (JAKOŚĆ SIĘ NIE ZMIENIA)
30	<ANY>	13.584543680898175	RESZTA ZDANIA (JAKOŚĆ SIĘ NIE ZMIENIA)

Kryterium dla decyzji obecności słowa w zdaniu

Dla każdej sekwencji wyznaczono najlepszą ścieżkę dla słowa „oś” i otrzymano w ten sposób jej jakość i długość. Jakość poprawnej ścieżki powinna być bliska zeru (brak kary za długość). Wariancja jakości powinna być zbliżona do wariancji jakości w oryginalnym słowie. Na tej podstawie wyznaczamy wartość dystrybuanty jakości słowa. Posiadając również rozkład średniej długości poprawnych sekwencji obserwacji dla słowa obliczamy gęstość prawdopodobieństwa dla długości znalezionej ścieżki.

W ten sposób ustalamy kryterium obecności słowa w postaci:

$$F(\text{jakość}) * f(\text{długość}) > 0.01$$

Przykład 2

Spróbujemy znaleźć słowo „oś” w 375 sekwencjach słów (zdaniach). Słowo to występuje w 285 sekwencjach słów (zdaniach).

Wyniki dla zdań zawierających słowo „oś”:

Próbka	**Zdanie**	**Decyzja według kryterium**	**F(jakość) * f(długość)**
0012001301.pfo	oś pięć kont zero jeden pięć	YES	0.01108565019249002
0012001302.pfo	oś pięć kont zero jeden pięć	YES	0.03303416812274212
0012001303.pfo	oś pięć kont zero jeden pięć	YES	0.01932385919468142
0012001304.pfo	oś pięć kont zero jeden pięć	YES	0.01870622236579566
0012001305.pfo	oś pięć kont zero jeden pięć	YES	0.01145780790982989
0012001306.pfo	oś pięć kont zero jeden pięć	YES	0.03214705044513515
0012001307.pfo	oś pięć kont zero jeden pięć	NO	0.00725564443378598
0012001308.pfo	oś pięć kont zero jeden pięć	YES	0.05066802535063948
0012001309.pfo	oś pięć kont zero jeden pięć	YES	0.07072412479249485
0012001310.pfo	oś pięć kont zero jeden pięć	YES	0.01172293383421189
0012001311.pfo	oś pięć kont zero jeden pięć	YES	0.07067790098726986
0012001312.pfo	oś pięć kont zero jeden pięć	YES	0.02341407946049514
0012001313.pfo	oś pięć kont zero jeden pięć	YES	0.04157817232722561
0012001314.pfo	oś pięć kont zero jeden pięć	YES	0.03671999155433262
0012001315.pfo	oś pięć kont zero jeden pięć	YES	0.05939730099245315

Wyniki dla zdań nie zawierających słowa „oś”:

Próbka	Zdanie	Decyzja wg. kryterium	F(jakość) * f(długość)
0013000401.pfo	dziewięćdziesiąt trzy	NO	7.932823541857408E-6
0013000402.pfo	dziewięćdziesiąt trzy	NO	7.667064570912386E-6
0013000403.pfo	dziewięćdziesiąt trzy	NO	7.91051930934518E-6
0013000404.pfo	dziewięćdziesiąt trzy	NO	1.7228211682893928E-6
0013000405.pfo	dziewięćdziesiąt trzy	NO	1.4066513424221048E-6
0013000406.pfo	dziewięćdziesiąt trzy	NO	1.1612799649827263E-6
0013000407.pfo	dziewięćdziesiąt trzy	NO	1.1612799649827263E-6
0013000408.pfo	dziewięćdziesiąt trzy	NO	3.53957228134704E-6
0013000409.pfo	dziewięćdziesiąt trzy	NO	3.53957228134704E-6
0013000410.pfo	dziewięćdziesiąt trzy	NO	1.7228211682893928E-6
0013000411.pfo	dziewięćdziesiąt trzy	NO	5.296060669208146E-6
0013000412.pfo	dziewięćdziesiąt trzy	NO	5.296060669208146E-6
0013000413.pfo	dziewięćdziesiąt trzy	NO	5.296060669208146E-6
0013000414.pfo	dziewięćdziesiąt trzy	NO	1.4891784631839952E-6
0013000415.pfo	dziewięćdziesiąt trzy	NO	8.008202202007737E-6

Łączne wyniki wyglądają następująco:

- Zdanie zawierało słowo i kryterium spełnione: 257 (90,2%)
- Zdanie zawierało słowo i kryterium niespełnione: 28 (9,8%)
- Zdanie nie zawierało słowa i kryterium spełnione: 0 (0%)
- Zdanie nie zawierało słowa i kryterium niespełnione: 90 (100%)

Dla słowa „oś” współczynnik błędnych klasyfikacji wynosił 7,6% ((0+28)/375).

5. Rozpoznawanie zdań

W rozdziałach poprzednich opisano sposób tworzenia Ukrytych Modeli Markowa reprezentujących pojedyncze słowa lub zdania. Opisano również algorytmy Viterbiego i Bauma-Welcha, które pozwalają uczyć te modele i rozpoznawać za ich pomocą sekwencje obserwacji. Zadanie klasyfikacji zdania polega na wybraniu najlepiej dopasowanego fonetycznie modelu zdania do zadanej sekwencji obserwacji. Możemy do tego celu użyć wielu modeli, z których każdy będzie opisywał jedno zdanie, lub jednego modelu zawierającego wszystkie rozpatrywane w danej chwili zdania.

5.1 WIELE OSOBNYCH MODELI CZY JEDEN ZINTEGROWANY MODEL?

Podejście z wieloma modelami posiada następujące cechy:

- Modele są prostsze, a co za tym idzie algorytmy uczące i rozpoznające działają dla nich znacznie szybciej niż dla pojedynczego modelu,
- Przeszukiwanie wielu modeli można łatwo zrównoleglić,
- Uczenie i rozpoznawanie zajmuje mniej pamięci,
- Przeszukiwanie Viterbiego zwraca zestaw wyników dopasowania fonetycznego dla każdego zdania z osobna,
- Przeszukiwanie Viterbiego musimy uruchamiać wielokrotnie dla każdego zdania z osobna.

Podejście z jednym modelem posiada następujące cechy:

- Model jest skomplikowany, a co za tym idzie algorytmy uczące i rozpoznające działają dla niego znacznie wolniej niż dla modeli pojedynczych zdań,
- Przeszukiwanie Viterbiego jest trudno zrównoleglić,
- Uczenie i rozpoznawanie zajmuje więcej pamięci,
- Przeszukiwanie Viterbiego uruchamiamy raz dla wszystkich zdań,
- Przeszukiwanie Viterbiego zwraca jeden wynik dopasowania fonetycznego i jedną najlepszą ścieżkę.

Biorąc również pod uwagę, że przeszukiwanie Viterbiego ma złożoność $O(N^2*Obs)$ (ilość stanów * ilość stanów * ilość obserwacji) lepiej jest przeszukiwać wiele mniejszych modeli[11].

5.2 KRYTERIUM WYBORU ZDANIA

W przypadku stosowania algorytmu Viterbiego dla każdego modelu zdania dostajemy jakość odpowiadającą sekwencji klas fonetycznych. Wybieramy model, dla którego jakość dopasowania jest największa. Dodatkowym kryterium może być również, by jakość wybranego zdania była dużo lepsza niż pozostałych. W ten sposób wprowadzamy przedział pewności i jeśli są dwa zdania o podobnej jakości wynik jest nieokreślony. Kolejnym kryterium może być średnia jakość zdania. Jeśli wynik uzyskany dla nowej próbki jest dużo mniejszy od średniej, nie powinniśmy mu ufać. Średnią jakość możemy określać w procesie uczenia. Możemy potraktować jakość jako zmienną losową o rozkładzie normalnym i obliczyć jej średnią i wariancję.

Jakość zwracana przez algorytm Viterbiego przyjmuje wartości od minus nieskończoności do zera. Wynika to z faktu, że jest ona logarytmem naturalnym prawdopodobieństwa zaakceptowania zdania przez Ukryty Model Markowa. Możemy w łatwy sposób powrócić do oryginalnego prawdopodobieństwa podnosząc liczbę E do potęgi równej wyznaczonej jakości zdania. W

[11] Na przykład 3 modele po 5 stanów i 10 obserwacji. Jeden model wymaga (3 *5) * (3 * 5) * 10 = 2250 operacji. Dla 3 oddzielnych modeli mamy natomiast 3 * 5 * 5 * 10 = 750 operacji. Przetwarzanie jest wiec 3 razy szybsze.

praktyce okazuję się jednak, że dla długich zdań otrzymamy zawsze zerowe prawdopodobieństwo. Dzieje się tak z powodu ograniczonej precyzji zapisu liczb zmiennoprzecinkowych.
W testowej aplikacji zastosowano więc pewien kompromis podczas prezentowania wynikowej jakości. Zmieniono podstawę potęgi, by była niewiele większa od jedności (dokładnie: 1.003), dzięki czemu podniesienie jej do potęgi będącej jakością daje przejrzyste wyniki w szerokim zakresie wartości. Tak obliczana jakość przyjmuje wartości od 0 do 1 i zachowuje się tak samo jak oryginalne prawdopodobieństwo, ale jest bardziej czytelna dla człowieka.

5.3 TESTY DLA JEDNEGO MÓWCY

Przykład 1.
Użytkownik wypowiedział zdanie: „oś trzy kont zero jeden pięć”. System przeszukał bazę HMM (każde zdanie to oddzielny model) wyznaczając dla każdego modelu współczynnik zwracany przez przeszukiwanie Viterbiego. Baza zdań zawiera między innymi (zestaw WAT):

1. zero
2. jeden
3. dwa
4. trzy
5. cztery
6. pięć
7. sześć
8. siedem
9. osiem
10. dziewięć
11. start
12. stop
13. lewo
14. prawo
15. góra
16. dół
17. puść
18. złap
19. oś
20. chwytak
21. zero cztery
22. siedemnaście
23. czterdzieści dziewięć
24. siedemdziesiąt dwa
25. dziewięćdziesiąt trzy

W rezultacie testów otrzymaliśmy poprawny wynik rozpoznania zdania:

Model opisywał zdanie	**Jakość[12]**	**Zlogarytmowana jakość**
zero	0.69757396093336 = 1.003^{-120}	-120
złap	0.6897369202682555	-124
góra	0.6780312936527959	-129
prawo	0.6757676379449079	-130
stop	0.6739962466206397	-131
start	0.6733520097041832	-132
lewo	0.6731747195917451	-132
dwa	0.667088496497475	-135
zero cztery	0.6536636989769992	-141
chwytak	0.6377079952502595	-150
dół	0.6362509031793654	-150
oś	0.6240494200906263	-157
osiem	0.618810878464455	-160
cztery	0.6181829915895678	-160
siedemdziesiąt dwa	0.6170537679179402	-161
jeden	0.6093194301330214	-165
siedem	0.6071625098091538	-166
siedemnaście	0.5952668924616444	-173
dziewięćdziesiąt trzy	0.5940533990586299	-173
puść	0.5929715267975251	-174
trzy	0.5896827689758044	-176

[12] Im większa jakość, tym lepiej dopasowany fonetycznie model do zadanej sekwencji. Jakość mieści się w zakresie od zera do jeden.

Przykład 2.
Przyjmujemy kryterium rozpoznania zdania: aby uznać zdanie za rozpoznane ocena (jakość) najlepiej dopasowanego zdania z modelu musi być lepsza o 15 pkt. od jakości rozpoznania dla pozostałych modeli.
Obliczymy też dystrybuantę rozkładu normalnego zmiennej losowej dla wyznaczonej jakości. Wartość dystrybuanty powie nam jakie jest prawdopodobieństwo, że poprawna sekwencja będzie miała jakość gorszą od wyznaczonej. Zakładamy, że wartość dystrybuanty nie może być mniejsza niż 0.001 (co 1000 poprawny model będzie odrzucony). Jeśli wartość dystrybuanty i różnica jakości spełnia nasze założenia podejmujemy decyzje. W przeciwnym wypadku prosimy o powtórzenie zdania.
Dla 10 wypowiedzi zdania „czterdzieści dziewięć" otrzymaliśmy następujące rezultaty:

Plik	Wybrane zdanie	Jakość	O ile lepsza jakość od innych zdań	Dystrybuanta rozkładu jakości	Pewność	Poprawnie
0023000201.pfo	czterdzieści dziewięć	0.453493579572	7.49669925247	0.30069126991	false	true
0023000202.pfo	czterdzieści dziewięć	0.510260086399	19.6138108849	0.51804665116	true	true
0023000203.pfo	czterdzieści dziewięć	0.455867818738	18.6605712477	0.30900559188	true	true
0023000204.pfo	czterdzieści dziewięć	0.432261105923	27.7352084086	0.23124949994	true	true
0023000205.pfo	czterdzieści dziewięć	0.431552976801	20.6746444081	0.22909837912	true	true
0033000201.pfo	czterdzieści dziewięć	0.530581127644	39.8793159425	0.59811155374	true	true
0033000202.pfo	czterdzieści dziewięć	0.573781979219	44.0891921053	0.75189673853	true	true
0033000203.pfo	czterdzieści dziewięć	0.550406697057	27.556292389	0.6724605829	true	true
0033000204.pfo	czterdzieści dziewięć	0.579925789395	45.868377748	0.7709286774	true	true
0033000205.pfo	czterdzieści dziewięć	0.517762752316	48.061261754	0.5478686708	true	true

Zdanie było rozpoznane poprawnie za każdym razem, ale w jednym przypadku nie byliśmy pewni wyniku (za mały odstęp).

Przykład 3:
Zbadamy wpływ uczenia na wyniki rozpoznawania. W systemie zarejestrowano 34 modele zdań. Przygotowano 375 próbek, które opisywały po 15 wypowiedzi 25 zdań . Uczenie prowadzono w 3 fazach:

- po 5 próbek uczących dla każdego z 25 zdań,
- po 10 próbek uczących dla każdego z 25 zdań,
- po 15 próbek uczących dla każdego z 25 zdań.

Po każdym etapie uczenia przeprowadzano testy rozpoznawania wszystkich próbek. Wyniki wyglądają następująco:

Ilość próbek uczących	**Wiarygodne rezultaty**	**Właściwe rozpoznania**
5	279 (74,4%)	373 (99,46%)
10	348 (92,8%)	373 (99,46%)
15	364 (97,06%)	375 (100%)

Proces uczenia pozwolił wyeliminować większość niepewnych wyników oraz występujące błędy klasyfikacji. Warunkiem koniecznym jest tutaj dobra jakość klas fonetycznych. Dla kodów źle dobranych, w których występują częste przekłamania, uzyskanie podobnych wyników jest trudne. Najlepiej również, by ilość klas fonetycznych była zbliżona do ilości pod-fonemów.

5.4 TESTY DLA WIELU MÓWCÓW

Testy zostały podzielone na dwie części. W pierwszej będą zaprezentowane wyniki rozpoznawania wypowiedzi wielu mówców wykonywane na modelach wytrenowanych dla jednego mówcy. Klasy fonetyczne dla wszystkich mówców będą identyczne. W kolejnej części będą zaprezentowane wyniki rozpoznawania dla modeli wytrenowanych za pomocą wypowiedzi kilku mówców.

Uczenie na podstawie wypowiedzi jednego mówcy

Testy były prowadzone dla dwóch zestawów zdań.
Pierwszy zestaw zawierał wypowiedzi dziesięciu mówców, w tym pięć głosów męskich i żeńskich. W zestawie znajdowało się dwadzieścia pięć typów zdań. Wszyscy mówcy nagrali pięć razy każde zdanie. Razem w zestawie wykorzystywano zatem 1250 próbek. Nagrania pochodzą z projektu prowadzonego na Wojskowej Akademii Technicznej.

Drugi zestaw zawierał wypowiedzi 3 mówców nagrane za pomocą syntezatora mowy. Ze względu na tematykę tych wypowiedzi nazywane są próbkami LOT.

Testy próbek z WAT
W tej fazie testów nagrania kolejnych mówców z osobna służyły w procesie klasteryzacji, kwantyzacji i treningu modeli fonetycznych. Sprawdzana była jakość rozpoznawania dla pozostałych mówców, dla których nie prowadzono procesu uczenia.

Zestaw 25 zdań z WAT podano w poprzednim punkcie.
Dziesięciu mówców zostało oznaczonych numerami. Ich lista przedstawia się następująco:

1. 001 – głos męski
2. 002 – głos żeński
3. 003 – głos męski
4. 004 – głos męski
5. 005 – głos męski
6. 006 – głos męski
7. 018 – głos żeński
8. 020 – głos żeński
9. 025 – głos żeński
10. 028 – głos żeński

Tabela. Wyniki klasyfikacji, gdy uczono modelu jedynie jednym głosem męskim (zaznaczone na czerwono są wyniki klasyfikacji tego głosu):

Klasyfikacja / Uczenie	001m 1-5	003m 1-5	004m 1-5	005m 1-5	006m 1-5	Podsumowanie głosy męskie - średnio
001 męski 1-5	90,8%	48,8%	66,4%	72,8%	65,6%	72,5%
002 kobiecy 1-5	33,6%	31,2%	25,6%	38,4%	60,8%	37,2%
003 męski 1-5	57,6%	100,0%	75,2%	81,6%	76,0%	74,7%
004 męski 1-5	72,8%	83,2%	99,2%	76,8%	82,4%	81,2%
005 męski 1-5	63,6%	72,0%	73,6%	100,0%	68,0%	73,5%
006 męski 1-5	58,0%	68,8%	64,8%	70,4%	99,2%	69,9%
018 kobiecy 1-5	49,6%	49,6%	48,8%	50,4%	57,6%	50,9%
020 kobiecy 1-5	64,8%	64,0%	67,2%	69,6%	70,4%	66,8%
025 kobiecy 1-5	17,2%	17,6%	12,8%	23,2%	34,4%	20,4%
028 kobiecy 1-5	34,0%	44,8%	28,8%	44,0%	48,0%	38,9%

Tabela. Wyniki klasyfikacji, gdy uczono modelu tylko jednym głosem żeńskim:

Klasyfikacja / Uczenie	002k 1-5	018k 1-5	020k 1-5	025k 1-5	028k 1-5	Podsumowanie głosy żeńskie
001 męski 1-5	47,2%	60,8%	63,2%	51,2%	34,4%	51,4%
002 kobiecy 1-5	96,0%	64,0%	56,0%	38,4%	31,2%	57,1%
003 męski 1-5	49,6%	66,4%	68,8%	43,2%	42,4%	54,1%
004 męski 1-5	54,4%	65,6%	73,6%	48,8%	40,0%	56,5%
005 męski 1-5	46,4%	56,8%	58,4%	48,8%	48,0%	51,7%
006 męski 1-5	63,2%	66,4%	70,4%	54,4%	44,0%	59,7%
018 kobiecy 1-5	58,4%	99,2%	76,0%	68,0%	68,8%	74,1%
020 kobiecy 1-5	52,8%	82,4%	98,4%	58,4%	50,4%	68,5%
025 kobiecy 1-5	35,2%	44,8%	35,2%	90,4%	28,8%	46,9%
028 kobiecy 1-5	44,0%	72,0%	60,8%	58,4%	100,0%	67,0%

Testy próbek LOT

Tabela. Uczenie kodami jednego mówcy. Klasyfikacja próbek wszystkich mówców:

Klasyfikacja / Uczenie	Michał 1-5	Michał 6-10	Sylwia 1-5	Sylwia 6-10	Olga 1-5	Olga 6-10	Średnia dla wszystkich
Michał 1-5[13]	100%	98,8%					99,4%
Michał 1-5	98,6%		40,6%		7,8%		49%
Sylwia 1-5	39,7%		99,8%		58,5%		65,9%
Olga 1-5	3,2%		23%		97,2%		41,1%

Uczenie na podstawie wypowiedzi i słownika kodowego wielu mówców

Wyniki ulegają znacznej poprawie wtedy, gdy już na etapie tworzenia kodów miesza się ze sobą w zestawie uczącym wielu mówców.
W tym podejściu podczas kodowania wygenerowano klastry i słownik kodowy na podstawie części wypowiedzi głosów żeńskich lub męskich. Następnie za pomocą takiego zestawu kodów przeprowadzono kwantyzację wypowiedzi wszystkich mówców. W fazie uczenia modelu HMM modele słów i zdań uczone były wyłącznie na podstawie próbek, które posłużyły do wygenerowania klastrów.

Testy próbek WAT

Tabela: Uczenie na podstawie 5 głosów żeńskich (po 2 próbki uczące na zdanie na mówcę). Klasyfikacja próbek uczących i po 3 nowych próbek na zdanie i na mówcę:

Mówca	Próbki uczące	Sukcesy	Porażki	Skuteczność
001m	Nie	86	39	68,8%
003m	Nie	86	39	68,8%
004m	Nie	76	49	60,8%
005m	Nie	84	41	67,2%
006m	Nie	99	26	79,2%

Średnia dla mężczyzn:		431	194	69,0%
002k	Tak	48	2	99,5%
002k	Nie	67	8	85,7%
018k	Tak	49	1	98,0%
018k	Nie	68	7	90,7%
020k	Tak	49	1	98,0%
020k	Nie	72	3	96,0%
025k	Tak	49	1	98,0%
025k	Nie	66	9	88,0%
028k	Tak	49	1	98,0%
028k	Nie	63	12	84,0%
Średnia dla kobiet:		580	43	93,1%

Tabela: Uczenie na podstawie 5 głosów męskich (po 2 próbki uczące na zdanie na mówcę). Klasyfikacja próbek uczących i po 3 nowych próbek na zdanie i na mówcę:

Mówca	**Próbki uczące**	**Sukcesy**	**Porażki**	**Skuteczność**
001m	Tak	50	0	100,0%
001m	Nie	62	13	82,7%
003m	Tak	50	0	100,0%
003m	Nie	65	10	86,7%
004m	Tak	49	1	98,0%
004m	Nie	66	9	88,0%
005m	Tak	49	1	98,0%
005m	Nie	71	4	94,7%
006m	Tak	50	0	100,0%
006m	Nie	72	3	96,0%
Średnia dla mężczyzn:		336	39	89,6%
002k	Tak	95	30	76,0%
018k	Tak	98	27	78,4%
020k	Tak	108	17	86,4%
025k	Tak	77	48	61,6%
028k	Tak	67	58	53,6%
Średnia dla kobiet:		445	150	74,8%

Uczenie na podstawie wypowiedzi wielu mówców (optymalizacja kodowania zgodnie z pkt. 4.2).

Tabela: Uczenie na podstawie 5 głosów żeńskich (kody wytworzone na podstawie tych mówców). Stosowano po 3 próbki uczące na zdanie na mówcę. Klasyfikacja próbek uczących i po 2 nowych próbek na zdanie i na mówcę:

Mówca	**Próbki uczące**	**Sukcesy**	**Porażki**	**Skuteczność**
002k	tak	73	2	97,3%
002k	nie	48	2	96,0%
018k	tak	73	2	97,3%
018k	nie	48	2	96,0%
020k	tak	74	1	98,7%
020k	nie	47	3	94,0%

025k	tak	71	4	94,7%
025k	nie	47	3	94,0%
028k	tak	73	2	97,3%
028k	nie	48	2	96,0%
Łącznie		602	23	96,3%

Tabela: Uczenie na podstawie głosów męskich (2 wariant kodów) (kody wytworzone na podstawie tych mówców). Stosowano po 3 próbki uczące na zdanie na mówcę. Klasyfikacja próbek uczących i po 2 nowych próbek na zdanie i na mówcę:

Mówca	**Próbki uczące**	**Sukcesy**	**Porażki**	**Skuteczność**
001m	tak	123	2	98,4%
001m	nie	117	8	93,6%
003m	tak	75	0	100,0%
003m	nie	42	8	84,0%
004m	tak	73	2	97,3%
004m	nie	43	7	86,0%
005m	tak	73	2	97,3%
005m	nie	48	2	96,0%
006m	tak	75	0	100,0%
006m	nie	50	0	100,0%
Łącznie		300	25	92,3%

Wyniki obu sposobów kodowania różnią sie tylko nieznacznie – przy nieoptymalizowanym kodowaniu średnia skuteczność klasyfikacji wynosiła 90-93%, a przy optymalizowanym 92-96%.

Testy próbek LOT

Dy dyspozycji były tylko 3 głosy, więc w obu krokach uczenia - kwantyzacji i uczenia modelu HMM mieszano w głosy żeńskie (5 próbek uczących i 5 nowych próbek):

Klasyfikacja / Uczenie	Michał 1-5	Michał 6-10	Sylwia 1-5	Sylwia 6-10	Olga 1-5	Olga 6-10	Średnia
Olga 1-5 oraz Sylwia 1-5			100%	98,0%	99,6%	97,2%	98,3%

6. Podsumowanie. Wnioski.

Opracowano algorytmy rozpoznawania słow kluczowych i zdan izolowanych w sygnale mowy. Wykonano implementacje obu algorytmów pozwalające na badanie sygnałów mowy ich testowanie, optymalizowanie ustawień parametrów tych algorytmów i poszukiwanie lepszych rozwiązań niektórych etapów analizy sygnału mowy.
Najpierw badano algorytmy dostosowane do rozpoznawania wypowiedzi jednego mówcy i jednego mikrofonu, tzn. wymagający osobnego trenowania dla każdego mówcy i sprzętu użytego do akwizycji sygnału mowy. Skuteczność rozpoznawania znacznie spadnie, jeśli wystąpią duże zmienności czasu trwania tej samej wypowiedzi lub istotnej zmiany intonacji tego samego zdania, np. zamiast intonacji oczekiwanej dla zdania informującego wystąpi intonacja dla pytania.
W drugiej kolejności badano algorytmy pod kątem rozpoznawania wypowiedzi wielu mówców i przy korzystaniu z różnych mikrofonów.

Algorytm rozpoznawania mowy jest niezależny od języka – wymaga jedynie wstępnego skonfigurowania modelu przez:
1. podanie zestawu fonemów języka i ich przynależności do 7 ogólnie przyjętych kategorii fonetycznych
2. podanie opisów rozpoznawanych zdań i słów kluczowych w postaci sekwencji wcześniej zdefiniowanych fonemów.

Pomimo korzystania z dość powszechnie w środowisku fachowym znanych metod podawanych w literaturze przedmiotu, opracowano algorytm rozpoznawania mowy o autorskim charakterze, modyfikując etapy analizy sygnału mowy na swój własny sposób, takie jak: modyfikacja algorytmu uczenia Bauma-Welcha i przeszukiwania Viterbiego w celu ich stosowania dla rozpoznawania słów i radzenia sobie z anomaliami zbyt krótkich i zbyt długich ścieżek.

Algorytmy umożliwiają zarówno rozpoznawanie całych zdań, jak i wyszukiwanie zadanych słów w zdaniu. Do obu zadań stosuje się zasadniczo te same funkcje uczenia i przeszukiwania przestrzeni hipotez. Zmienia sie jedynie sposób oceny wiarygodności ścieżki rozwiązania. Wynika to z różnicy polegającej na tym, że w rozpoznawaniu zdań oczekiwaną długość ścieżki wyznacza wektor obserwacji a w rozpoznawaniu słów – oczekiwana długość ścieżki modelującej dane słowo

Przedstawiono wyniki eksperymentów przeprowadzone na dwóch zestawach danych (WAT i LOT) o różnych liczbach zdań (klas) (jednorazowo klasyfikowano maksymalnie 25 typów zdań dla WAT i 50 dla LOT). Proces uczenia prowadzono na jednej połowie (lub części) zestawu danych, a proces klasyfikacji i rozpoznawania – na całości zestawu.

W warunkach uczenia i klasyfikacji pojedynczego mówcy skuteczność rozpoznawania wynosiła:
- dla nowych próbek - ponad 90%,
- a dla próbek uczących zwykle sięgała 98-100%.

Podsumowanie wyników rozpoznawania z modelem HMM

1. Zastosowana dyskretna postać prawdopodobieństw wyjść (obserwacji) w modelu HMM sprawia, że potrzebna jest stosunkowa duża liczba próbek uczących. Dzięki temu wszystkie kody słownika kodowego pojawią się ze statystyką dobrze przybliżającą rzeczywiste prawdopodobieństwa ich wystąpień.

2. Uczenie modelu HMM jedynie na podstawie kodów jednego mówcy powoduje znikomą skuteczność rozpoznawania innych głosów poza tym jednym, dostępnym podczas uczenia (pkt. 1.1 i 2). Dla próbek LOT skuteczność rozpoznawania próbek głosu „uczącego" wyniosła 95%, 92% i 61%. Skuteczność dla innych głosów, nie znanych w fazie uczenia, była bardzo niska. Dla próbek LOT wynosiła ok. 21-26%, gdy głos był tej samej płci co głos „uczący" lub 5%, gdy był innej płci. Dla próbek WAT próbki innych głosów, poza uczącym, rozpoznawane były ze skutecznością od ok. 20% do 50%.

3. Uczenie modelu HMM na podstawie kodów różnych mówców, nawet wtedy, gdy słownik kodowy stworzono na podstawie jedynie jednego głosu, znacznie poprawia skuteczność rozpoznawania (pkt. 1.2). Dla próbek WAT przeciętna skuteczność rozpoznania wszystkich próbek wszystkich 10 mówców rosła od 65 % (uczenie próbkami kodów 2 kobiet), 70% (uczenie kodami próbek 3 kobiet), 76% (uczenie próbkami kodów 3 kobiet i 1 mężczyzny) do 80% (dla 3 kobiet i 2 mężczyzn).

4. Najlepsze wyniki rozpoznawania uzyskuje się stosując słowniki kodowe wyznaczone w oparciu o próbki wielu mówców, a nie tylko jednego, jak to było w poprzednich omawianych przypadkach. Dla próbek WAT zastosowanie słownika kodowego głosów żeńskich i uczenie modelu HMM częścią próbek głosów żeńskich sprawia, że skuteczność rozpoznawania wynosi od 84% do 100% dla głosów żeńskich a dla męskich: 55-79%.

Wnioski

1. Proponujemy korzystanie z dwóch odrębnych słowników kodowych cech akustycznych, utworzonych na podstawie próbek kilku osób:
    a. kobiet (F0: 200 – 250 Hz, średnia DP (dolnopasmowość): 0.1-0.5) lub
    b. mężczyzn (F0: 100 – 150 Hz, śDP: 0.5-0.9).
2. Uczenie początkowe należy przeprowadzić na podstawie próbek kilku osób jednocześnie. Przy korzystaniu z programu przez jedną osobę warto zastosować „adaptowanie się" się słownika kodowego dla modelu HMM do próbek głosu danej osoby.
3. Dla wystarczająco dużej liczby próbek uczących klasyfikator HMM powinien osiągnąć skuteczność prezentowaną przez klasyfikator DTW, pomimo, że korzysta on ze znacznie zredukowanej informacji o sygnale (dyskretne kody) w porównaniu z klasyfikatorem DTW (wielowymiarowe wektory cech). Możliwość tworzenia osobnych modeli fonemów i słów w klasyfikatorze opartym o HMM pozwala na łatwą rozszerzalność (bez dodatkowych próbek uczących) modelu dla dowolnych zdań złożonych ze słów pochodzących ze znanego słownika. Jest to zasadnicza zaleta klasyfikatora HMM nad DTW i dlatego dla dużych zbiorów zdań powinien być tworzony klasyfikator HMM w miejsce DTW.

Literatura

[ADA00] B. Adamczyk, K. Adamczyk, K. Trawiński: *Zasób mowy ROBOT*. Biuletyn IAiR nr 12, Wojskowa Akademia Techniczna, Warszawa, 2000 (+2 CD-romy).

[BEN08] J. Benesty, M.M. Sondhi, Y. Huang (Eds.): *Springer Handbook of Speech Processing*. Springer-Verlag Berlin Heidelberg, 2008.

[CSL00] (Praca zbiorowa). *The CSLU Speech Toolkit*. Centre for Spoken Language Understanding, Oregon Graduate Institute of Science and Technology, USA, 2000.

[HTK06] P. Woodland, G. Evermann, M. Gales: HTKBook, Cambridge University Engineering Department (CUED), 2000-2006, http://htk.eng.cam.ac.uk

[INT10] InteliWise SA: Zestaw próbek LOT. Warszawa, 2010. (informacja prywatna)

[KAS09] W. Kasprzak. *Rozpoznawanie obrazów i sygnałów mowy*. Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa, 2009. http://www.wydawnictwopw.pl/

[KAS12] W. Kasprzak: *Klasyfikacja zdań w sygnale mowy z wykorzystaniem modelu HMM*. Raport badawczy. IAiIS 12-04, Instytut Automatyki i Informatyki Stosowanej Politechnikmi Warszawskiej, 2012.

[RAB93] L. Rabiner, B. Juang. *Fundamentals of Speech Recognition*, Prentice Hall, 1993.

[WAL04] W. Walker, P.Lamere, P. Kwok, B. Raj, R. Singh, E. Gouvea, P. Wolf, J. Woelfel. *Sphinx-4: A Flexible Open Source Framework for Speech Recognition*. SMLI TR2004-0811, SUN Microsystems Inc., 2004. http://cmusphinx.sourceforge.net/sphinx4/#whitepaper

[WYD09] S. Wydra. *Zastosowanie ukrytych modeli Markowa w aplikacjach głosowych dla mowy polskiej*. Rozprawa doktorska. Wydział Elektroniki i Technik Informacyjnych Politechniki Warszawskiej, Warszawa 2009.