ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1999
Seria: INFORMATYKA z. 37 Nr kol. 1421

Krzysztof CYRAN, Tomasz PODESZWA
Politechnika Śląska, Instytut Informatyki

ZASTOSOWANIE HMM i NN DO ROZPOZNAWANIA KONTEKSTOWEGO W PRZETWARZANIU MOWY

Streszczenie. Artykuł dotyczy problematyki rozpoznawania zależnego od kontekstu, na przykładzie systemów rozpoznawania i generowania mowy. Przedstawiono w nim zastosowanie sztucznych sieci neuronowych i ukrytych modeli Markowa w strukturze takich systemów. Zwrócono uwagę na potencjalne możliwości przyśpieszenia rozpoznania w systemach z sieciami neuronowymi po wykorzystaniu rozkazów technologii MMX.

USE OF HMM AND NN FOR CONTEXT DEPENDENT RECOGNITION IN SPEECH PROCESSING

Summary. The article concerns the problem of context dependent recognition tasks on the basis of speech recognition and speech generation systems. We present use of ANNs and HMMs in the structure of such systems. We point out the potential possibility of speed-up the recognition in systems with ANNs after utilization of MMX technology commands.

1. Przetwarzanie mowy jako rozpoznawanie kontekstowe

Z rozpoznawaniem kontekstowym mamy do czynienia wszędzie tam, gdzie klasyfikacja wektora wejściowego $X = [x_1, x_2, \ldots, x_n]$ określonego w $\Re^n$ jest uzależniona od dodatkowego wektora wejściowego zwanego kontekstem $A = [\alpha_1, \alpha_2, \ldots, \alpha_m]$ w dziedzinie $\Re^m$. Przestrzeń kontekstu $\Re^m$ w problemach powiązanych z czasem z reguły można rozpatrywać jako złożenie dwóch podprzestrzeni: podprzestrzeni kontekstu lewostronnego (czas ujemny) $\Re^s$ oraz podprzestrzeni kontekstu prawostronnego (czas dodatni) $\Re^t$. Spełnione muszą być oczywiście:

$$\Re^m = \Re^s \times \Re^t \tag{1}$$

$$m = s + t \tag{2}$$

Wtedy wektor kontekstu $A = [\alpha_1, \alpha_2, \ldots, \alpha_m]$ można traktować jako konkatenację wektora kontekstu lewostronnego $A_l = [\alpha_{l1}, \alpha_{l2}, \ldots, \alpha_{ls}]$ i wektora kontekstu prawostronnego $A_p = [\alpha_{p1}, \alpha_{p2}, \ldots, \alpha_{pt}]$. Tak określone rozpoznawanie można przedstawić jako wektorową funkcję f odwzorowującą $\Re^n \times \Re^m \rightarrow \Re^k$, taką że:

$$Y = [y_1, y_2, \ldots, y_k] = f(X, A) \Leftrightarrow y_i = P(c_i | X, A), \forall i \tag{3}$$

gdzie: $P(c_i|X,A)$ oznacza prawdopodobieństwo, że wektor wejściowy X znajdujący się w kontekście A należy rozpoznać jako klasę c_i. Łatwo wykazać, że zarówno generowanie, jak rozpoznawanie mowy stanowi klasę tego typu zadań, w których problemem pozostaje wyznaczenie nie tylko funkcji f, ale również rozmiarów przestrzeni kontekstu, po to by realizowane przez system przybliżenie funkcji dawało niewielkie błędy. W przypadku rozpoznawania mowy kontekst niezbędny jest ze względu na to, że sąsiednie dźwięki wpływają poprzez koartykulację na inne, natomiast przy generacji mowy również przejście litera → fonem uzależnione jest od otoczenia tejże litery. Wynika stąd, że generacja mowy też jest procesem rozpoznawania, gdzie zarówno wektor X, jak i A określone są dla zbioru liter, a wektor Y składa się z klas odpowiadających fonemom, sylabom, słowom w zależności od ziarnistości przyjętego modelu akustycznego. W dalszej części przedstawione zostaną systemy przetwarzania oparte na ukrytych modelach Markowa HMM (ang. Hidden Markov Models) i sieciach neuronowych NN (ang. Neural Networks) oraz ich hybrydy, jednak ze względu na podstawowy fakt, że mowa jest informacją dźwiękową, najpierw zaprezentujemy w skrócie kanały akustyczne we współczesnych mikrokomputerach, ze szczególnym uwzględnieniem tych zagadnień, które mają bezpośredni wpływ na szybkość procesu rozpoznawania.

2. Kanały akustyczne w mikrokomputerach

Dzisiejsze mikrokomputery, określane mianem multimedialnych, do komunikacji z człowiekiem (oprócz obrazu) wykorzystują informację dźwiękową. Budowie elementów odpowiadających za możliwości sprzętowe i programowe najniższego poziomu w systemach przetwarzających mowę poświęcony jest niniejszy rozdział.

2.1. Reprezentacja informacji dźwiękowej

Sygnał dźwiękowy jest przedstawiany jako zmiana ciśnienia akustycznego w czasie. W celu dowolnego, cyfrowego przetwarzania takiego sygnału niezbędne jest jego próbkowanie i kwantyzacja. Ponieważ dla typowego sygnału mowy zakres częstotliwości wynosi ok. 8kHz (dla sygnału telefonicznego 4 kHz), zatem aby przenieść całą informację zawartą w sygnale,

zgodnie z twierdzeniem Shannona, należy próbkować go z częstotliwością 16kHz (dla telefonicznych rozmów wystarczy 8kHz). W ogólności zatem dostajemy około 16000 wartości z każdej sekundy mowy, co jest dość znaczącą ilością, jeśli chcemy przetwarzać dłuższe wypowiedzi, a nie zastosujemy żadnej metody kompresji.

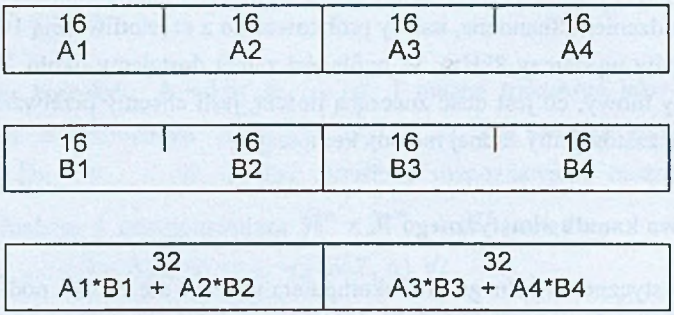
2.2. Budowa kanału akustycznego

Kanały akustyczne dowolnego mikrokomputera składają się z kilku podstawowych układów. Należą do nich przetworniki A/C oraz C/A, dla których określona jest maksymalna częstotliwość próbkowania i odtwarzania, znajdujące się w układzie DSP (Digital Sound Processor). W najprostszych wersjach układ ten jest tylko w stanie odebrać bądź odtworzyć kwant informacji dźwiękowej, bardziej rozbudowane układy DSP potrafią go jeszcze sprzętowo przetworzyć. W komputerach typu PC kanały akustyczne reprezentowane są poprzez karty dźwiękowe, najczęściej zgodne ze standardem SoundBlaster. Karty te posiadają dodatkowo syntetyzer FM, wykorzystywany przy odtwarzaniu muzyki, jednak zbyt prymitywny, by mógł pozwolić na sztuczną syntezę ludzkiej mowy. Dlatego do generowania mowy stosowane są programy syntezy, np. formantowy, szeregowo-równoległy syntezator mowy pracujący według systemu Klatta [5]. Podobnie, rozpoznawania mowy nie dokonuje się, w większości przypadków, na poziomie sprzętu - w każdym razie nie na wszystkich etapach tego procesu. Podstawą zatem każdego systemu przetwarzającego informację dźwiękową jest stworzenie interfejsu, poprzez który wyższe warstwy systemów rozpoznawania, bądź generacji mowy, mogłyby komunikować się z układem DSP.

Taki interfejs niskiego poziomu, służący do komunikacji z układem DSP dla karty SoundBlaster został przez nas zaprojektowany i zaimplementowany w C++.

2.3. Technologia MMX

Nowa generacja procesorów firmy Intel wyposażona jest w zestaw dodatkowych 57 rozkazów, które zaprojektowano w celu przyśpieszenia operacji związanych z przetwarzaniem przede wszystkim obrazu, ale również dźwięku. Szczególne znaczenie ma tutaj rozkaz o nazwie PMADDWD (Packed Multiply and Add), umożliwiający w jednym cyklu zegarowym wykonanie mnożenia czterech słów argumentu 64-bitowego z czterema innymi słowami drugiego argumentu, a następnie zsumowanie pierwszych dwóch iloczynów oraz pozostałych dwóch iloczynów i zapisanie wyników w dwóch podwójnych słowach rejestru docelowego MMX (rys. 1).



Rys. 1. Działanie rozkazu MMX PMADDWD
Fig. 1. Results of MMX PMADDWD command

Wyżej przedstawiona operacja jest wykorzystywana w obliczaniu iloczynu skalarnego dwóch wektorów. Z drugiej strony można zauważyć, że najbardziej czasochłonna czynność obliczania pobudzania sieciowego net_i, dla i-tego węzła sieci neuronowej NN jest realizowana właśnie jako iloczyn skalarny wektora wag $w_i = [w_{i1}, w_{i2}, \ldots, w_{ik}]$ dochodzących do tego węzła, oraz wektora pobudzeń wejściowych wygenerowanych przez poprzednią warstwę $O = [O_1, O_2, \ldots, O_k]$, zgodnie ze wzorem:

$$net_i = \sum_j w_{ij} O_j \tag{4}$$

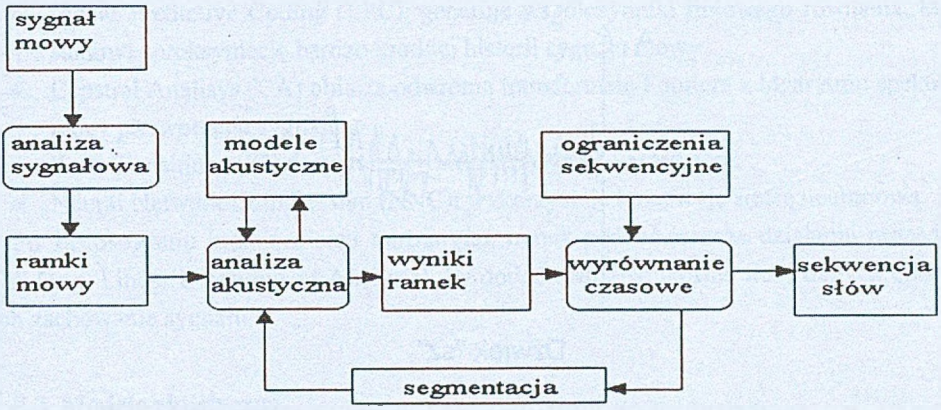
Ponieważ architektury NN są, obok ukrytych modeli Markowa HMM, techniką najczęściej wykorzystywaną do celów rozpoznawania dźwięków, zatem zastosowanie rozkazów MMX powinno w znaczący sposób wpłynąć na skrócenie czasu realizacji tych zadań. W rozkazach MMX jednoczesne wykonanie tak wielu operacji możliwe jest dzięki zastosowaniu techniki SIMD (single instruction, multiple data) i według firmy Intel prowadzi do pięciokrotnego przyśpieszenia operacji sumowania iloczynów. Od oprogramowania wykorzystującego te rozkazy można więc oczekiwać prawie takiego samego przyśpieszenia symulacji NN.

3. Rozpoznawanie mowy

W dotychczasowych pracach nad rozpoznawaniem mowy nie zbudowano jeszcze ogólnego modelu, którego implementacja przy dzisiejszych środkach obliczeniowych dawałaby system niezależny od mówcy, przystosowany do rozpoznawania dowolnych wypowiedzi mowy ciągłej. Dlatego, obok istniejących, klasycznych rozwiązań systemów rozpoznawania mowy z wykorzystaniem HMM, podejmowane są próby wykorzystania innych metod, np. poprzez zastosowanie NN. Zainteresowanie systemami NN motywowane jest potwierdzoną wieloma eksperymentami (również naszymi) przydatnością tychże do celów rozpoznawania kontekstowego.

Jak wspomniano we wstępie, informacja dźwiękowa w swej nieprzetworzonej formie zajmuje około 16000 próbek/sekundę. Jest to dużo, ale nie jest to jedyny problem, na jaki natrafiamy chcąc rozpoznać mowę ludzką. Jeżeli przyjrzymy się wykresom zmian amplitudy ciśnienia akustycznego w czasie dla tych samych słów wypowiadanych przez różne osoby, z których każda ma inną barwę dźwięku, mówi z inną wysokością lub inaczej dane słowo intonuje, okazuje się, że wykresy te przedstawiają się bardzo różnie i niezwykle trudno jest rozpoznać z takiego wykresu, że w istocie wszystkie te reprezentacje odnoszą się do tego samego słowa. Kolejną trudnością w przypadku rozpoznawania mowy ciągłej jest fakt, że z wykresu zależności amplitudy od czasu nie da się wnioskować o podziale danej wypowiedzi na słowa. Segmentacji sygnału mowy na słowa nie da się jednoznacznie przeprowadzić na podstawie tylko fizycznych wielkości reprezentujących sygnał mowy (choć podejmowane są próby wykorzystujące informacje energetyczne o sygnale). Do tego niezbędna jest także pewna wiedza o zasobach językowych poszczególnych języków, która musi być zaszyta w systemie rozpoznawania mowy. Kolejną trudność, tym razem na poziomie najmniejszych jednostek, jakie człowiek potrafi słuchowo wydzielić z sygnału mowy, czyli fonemów, stanowi fakt, że dźwięki, które rozpoznajemy jako te same fonemy, w rzeczywistości różnią się dość znacznie, w zależności od tego jakie inne dźwięki z nimi sąsiadują (zjawisko koartykulacji). Wszystko to razem sprawia, że rozpoznawanie mowy ciągłej, niezależne od użytkownika, dla wszystkich słów danego języka nie zostało dotąd rozwiązane i stanowi wciąż jeszcze zbyt ambitne zadanie, choć - jako cel długofalowy systemów rozpoznających mowę - wydaje się jak najbardziej naturalne. Ze względu na złożoność tych zagadnień w naszej pracy zajęto się rozpatrzeniem tylko niektórych faz składających się na cały proces.

3.1. Architektura systemu rozpoznawania mowy



Rys. 2. Struktura typowego systemu rozpoznawania mowy
Fig. 2. Classical speech recognizer architecture

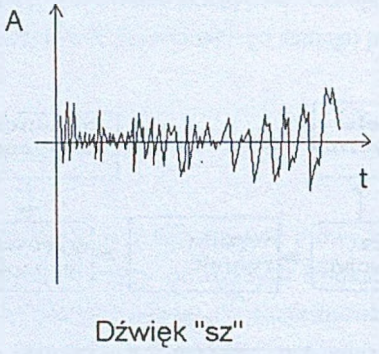
Ogólna struktura systemu rozpoznawania mowy przedstawiona jest na rysunku 2. Wyróżnić w niej można następujące zadania do zrealizowania:

- Analiza sygnałowa
- Analiza akustyczna
- Wyrównanie czasowe
- Segmentacja

Dopiero w wyniku przetworzenia sygnału mowy przez te procesy na wyjściu pojawi się prawidłowa sekwencja rozpoznanych słów.

3.2. Analiza sygnałowa

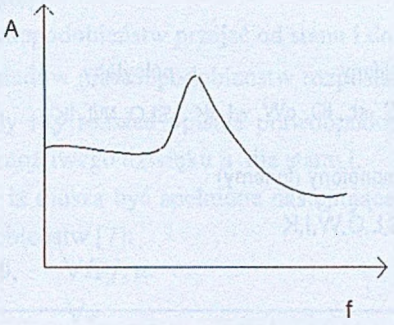
Sygnał mowy, zanim zostanie poddany bardziej wyrafinowanym środkom, zmierzającym do jego interpretacji, powinien zostać wstępnie przetworzony na postać różną od tej, którą otrzymujemy poprzez zwykłe jego próbkowanie z częstotliwością 16 kHz. Istnieje wiele możliwych sposobów takiego wstępnego przetworzenia tego sygnału, m.in. transformacja Fouriera. Jednym z głównych argumentów za przyjęciem tej metody jest fakt, że ma ona swój pierwowzór w naturze. Budowa ucha ludzkiego powoduje, że na tzw. błonie podstawowej dokonuje się proces mający z grubsza biorąc charakter analizy Fouriera [14]. Dzięki temu na zasadzie analogii możemy próbować rozpoznawać mowę w sposób sztuczny, wykorzystując wzorce z natury. Daje nam to gwarancję, że w przekształceniu nie zagubimy informacji użytecznej do rozpoznania. Ponadto, stosując tę transformację, około dziesięciokrotnie zmniejsza się ilość przetwarzanych wielkości przypadających na jednostkę czasu. Dość dobrze obrazują to rysunki 3 oraz 4, z których wyraźnie widać, że charakter zmian amplitudy dla poszczególnych częstotliwości jest o wiele łagodniejszy, niż charakter zmian amplitudy ciśnienia w zależności od czasu.



Rys. 3. Spróbkowany dźwięk „sz" w dziedzinie czasu
Fig. 3. Sampled sound „sh" in time domain

Zastosowanie transformacji daje nam zatem możliwość rzadszego próbkowania. W większości wypadków cały wymagany zakres częstotliwości można próbkować tylko dla 16 wybranych częstotliwości, jeżeli transformacji dokonywać będziemy na odcinkach czasu rzędu 10 ms. W wyniku tej operacji otrzymamy:

16 amplitud * 100ramek/s. = 1600 wartości na sekundę trwania sygnału mowy (zamiast początkowych 16000).



Rys. 4. Dźwięk „sz" w dziedzinie częstotliwości
Fig. 4. Sound „sh" in frequency domain

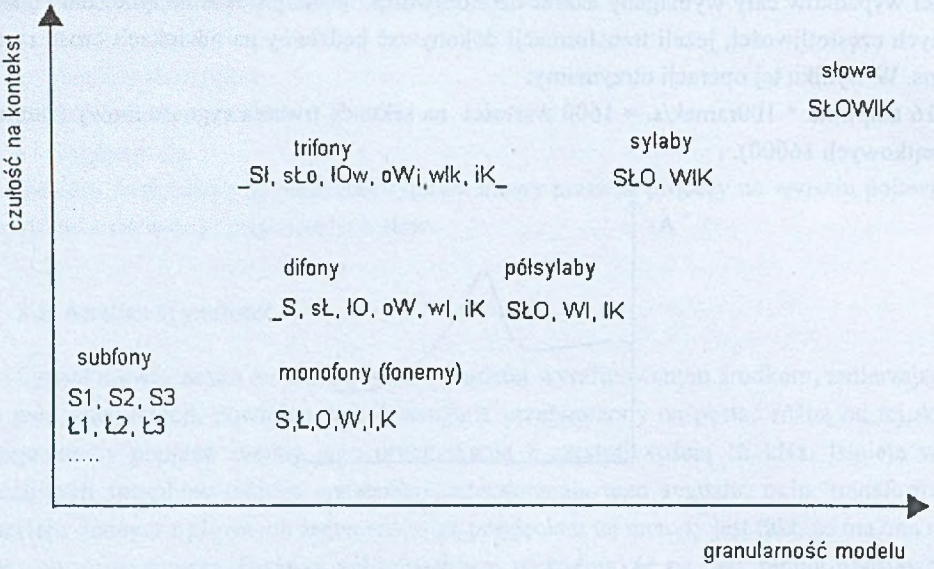
Poza transformatą Fouriera na tym etapie przetwarzania można stosować następujące metody [6, 7, 8], które dokonują ekstrakcji użytecznych w dalszej analizie cech, nie tracąc zbyt wiele użytecznej informacji:

- Perceptual Linear Prediction (PLP): to metoda też uwarunkowana fizjologią ucha, która wszakże generuje cechy, nie mające swojej graficznej interpretacji, jak w przypadku analizy Fouriera, której wynikiem był rozkład częstotliwości.
- Linear Predictive Coding (LPC): generuje współczynniki liniowego równania, które stanowi aproksymację bardzo krótkiej historii sygnału mowy.
- Cepstral Analisys (CA) oblicza odwrotną transformatę Fouriera z logarytmu spektrum mocy pierwotnego sygnału.
- Zero Transitions (ZT): zapamiętuje przejścia sygnału przez zero.
- Neural Network Compression (NNC): wykorzystuje kompresję siecią neuronową.

Po zastosowaniu którejś z nich można ciąg ramek poddać jeszcze działaniu procedury LDA (ang. Linear Discriminant Analysis), by dodatkowo zredukować ilość danych opisujących zachowanie sygnału.

3.3. Modele akustyczne

Kolejnym etapem jest analiza akustyczna, związana ściśle z przyjętymi modelami akustycznymi.



Rys. 5. Porównanie ziarnistości modeli akustycznych
Fig. 5. Comparison of acoustic model granularity

Na rys. 5. przedstawiono wykres zależności czułości na kontekst typowych modeli podstawowych w odniesieniu do ich ziarnistości. Im większa czułość na kontekst modelu, tym dokładniejsze będzie rozpoznawanie z użyciem tego modelu, o ile będzie dobrze wytrenowany. Niestety dla systemów o największej czułości kontekstowej z reguły wzrasta ziarnistość modelu, przez co nie daje się ich dobrze wytrenować przy dostępnej liczbie przykładów. Dlatego rzadko stosuje się modele słowowe, i sylabowe, których czułość na kontekst jest największa, lecz są one dużej ziarnistości. Bardziej obiecujące są modele akustyczne z jednostką trifonu, dla których ziarnistość jest porównywalna z modelem fonemowym, a czułość na kontekst o wiele większa. Zwłaszcza w podejściu z czystymi HMM duża czułość kontekstowa modelu akustycznego odgrywa znaczącą rolę, gdyż tam z założenia o niezależności wynika, iż można rozpatrywać tylko jedną ramkę mowy i jedynie na jej podstawie wyliczać prawdopodobieństwa wygenerowania poszczególnych symboli. Przy zastosowaniu architektur NN ogólną czułość na kontekst systemu można zwiększyć przez rozszerzenie okna przesuwnego na kilka ramek mowy, w których będzie zawarta informacja o kontekście analizowanej ramki. Dzięki temu przy zastosowaniu NN analizujących ok. 10 ramek równocześnie osiąga się wyniki porównywalne z użyciem modeli trifonowych w HMM, nawet jeśli przyjmie się prosty model monofonowy (fonemowy). Nie przekreśla to wszakże celowości użycia także i w tych systemach w systemach modelu difonowego lub nawet trifonowego. W pracy [3] podaje się także zalety modelu diafonowego. Przedstawione tam wyniki dla systemu roz-

poznającego 100 słów w trójwarstwowej NN dawały 10% błędu dla modelu diafonowego, i aż 18% błędu dla modelu monofonowego.

3.4. Ukryte Modele Markowa

Formalnie HMM ma następujące składniki:

{s} - zbiór stanów;

$\{a_{ij}\}$ - zbiór prawdopodobieństw przejść od stanu i do stanu j,

$\{b_i(u)\}$ - zbiór rozkładów prawdopodobieństw rozpostartych w przestrzeni akustycznej. Każdy i-ty rozkład opisuje prawdopodobieństwo wygenerowania dowolnego możliwego dźwięku u dla stanu i.

Z powyższego wynika, iż muszą być spełnione następujące równania, charakterystyczne dla dowolnych prawdopodobieństw [7]:

$$a_{ij} > 0, \quad b_i(u) > 0, \quad \forall i,j,u \tag{5}$$

$$\sum_j a_{ij} = 1, \quad \forall i \tag{6}$$

$$\sum_u b_i(u) = 1, \quad \forall i \tag{7}$$

Sposób reprezentowania rozkładu $b_i(u)$ decyduje o jednej z podstawowych słabości HMM, gdyż stosując rozkłady parametryczne (np. suma rozkładów Gaussa), z góry przesądzamy o kształcie powierzchni w przestrzeni akustycznej, natomiast rozkłady dyskretne (nieparametryczne) zawierają zawsze błędy kwantyzacji. NN potrafią odwzorowywać te rozkłady dokładniej i tym samym w tym punkcie mają teoretyczną przewagę nad HMM. Jednakże zanim pokażemy, jak stosować NN w systemach rozpoznawania mowy, przedstawimy algorytmy wykorzystywane przez HMM [7]. Są to:

- algorytm typu FORWARD, użyteczny przy rozpoznawaniu pojedynczych słów,
- algorytm Viterbi, do rozpoznawania mowy ciągłej,
- algorytm typu FORWARD-BACKWARD, stosowany do uczenia HMM.

W kolejnych rozdziałach algorytmy te zostaną pokazane szczegółowiej.

3.4.1. Algorytm forward

Jest to jeden z przykładów programowania dynamicznego. Jego złożoność czasowa oraz pamięciowa jest liniowa, a służy do obliczania prawdopodobieństwa, że dany HMM wygenerował sekwencję wyjściową $y_1^T = (y_1, y_2, \ldots, y_T)$.

Zdefiniujmy $\alpha_j(t)$ jako prawdopodobieństwo wygenerowania przez HMM częściowej sekwencji y_1^t, kończąc w stanie j w chwili t. $\alpha_j(t=0)=1$ w stanie początkowym oraz $\alpha_j(t=0)=0$ w pozostałych stanach. Prawdopodobieństwa dla kolejnych ramek dane są wzorem rekurencyjnym:

$$\alpha_j(t) = \sum_i \alpha_i(t-1) a_{ij} b_j(y_t) \tag{8}$$

Jeżeli F jest stanem końcowym, to przez indukcję wnioskujemy, że $\alpha_F(T)$ jest poszukiwanym prawdopodobieństwem, że HMM wygenerował pełną sekwencję wyjściową $y_1^T = (y_1, y_2, ..., y_T)$.

3.4.2. Algorytm Viterbi

Algorytm ten służy do wyznaczania najbardziej prawdopodobnej sekwencji stanów, która doprowadziła do wygenerowania $y_1^T = (y_1, y_2, ..., y_T)$. Dzięki temu możliwa będzie segmentacja ramek mowy, a co za tym idzie algorytm może być stosowany w rozpoznawaniu mowy ciągłej. Algorytm jest bardzo podobny do algorytmu Forward, a główna różnica polega na tym, że zamiast sumowania w każdej ramce, w algorytmie Viterbi obliczane jest maksimum:

$$v_j(t) = \underset{i}{MAX}\,[v_i(t-1) a_{ij} b_j(y_t)] \tag{9}$$

Maksimum to wyznacza implicite jeden najlepszy poprzedni stan. Należy teraz explicite zapamiętywać te poprzednie stany, by po osiągnięciu $v_F(T)$ w końcowym stanie na ostatniej ramce mowy, być w stanie odtworzyć całą sekwencję stanów. Z sekwencji stanów zaś prosto można odtworzyć sekwencję słów.

3.4.3. Algorytm uczenia forward-backward

Uczenie HMM polega na optymalizacji parametrów a oraz b. Optymalizacja taka nie może być dokonana na podstawie analitycznych obliczeń, natomiast jest możliwa przez iteracyjną reestymację zestawu tych parametrów. Zdefiniujmy $\beta_j(t)$ jako prawdopodobieństwo wygenerowania reszty sekwencji, tj. y_{t+1}^T, zaczynając od stanu j w chwili t. Wtedy:

$$\beta_j(t) = \sum_k \beta_k(t+1) a_{jk} b_k(y_{t+1}) \tag{10}$$

Dodatkowo, niech $\gamma_{ij}(t)$ będzie prawdopodobieństwem przejścia ze stanu i do stanu j w chwili t, zakładając, że całą sekwencję $y_1^T = (y_1, y_2, ..., y_T)$ wygenerował bieżący HMM.

Wtedy:

$$\gamma_{ij}(t) = P(i_t \to j | y_1^T) = \frac{P(i_t \to j, y_1^T)}{P(y_1^T)} = \frac{\alpha_i(t) a_{ij} b_j(y_{t+1}) \beta_j(t+1)}{\sum_k \alpha_k(T)} \tag{11}$$

Zdefiniujmy $N(i \to j)$ jako wartość oczekiwaną ilości tranzycji ze stanu i do j w czasie od 1 do T:

$$N(i \to j) = \sum_t \gamma_{ij}(t) \tag{12}$$

Sumując to po wszystkich stanach przeznaczenia j, otrzymamy $N(i \to *)$ oznaczające oczekiwaną liczbę odwiedzenia stanu i w czasie od 1 do T.

$$N(i \to *) = \sum_j \sum_t \gamma_{ij}(t) \tag{13}$$

Wybierając tylko te odwiedziny, kiedy stan i wyemitował symbol u, mamy $N(i,u)$:

$$N(i,u) = \sum_{t:(y_t=u)} \sum_j \gamma_{ij}(t) \tag{14}$$

Teraz można reestymować parametry, otrzymując ich nowe wartości jako:

$$\overline{a_{ij}} = P(i \to j) = \frac{N(i \to j)}{N(i \to *)} = \frac{\sum_t \gamma_{ij}(t)}{\sum_j \sum_t \gamma_{ij}(t)} \tag{15}$$

$$\overline{b_i(u)} = P(i,u) = \frac{N(i,u)}{N(i \to *)} = \frac{\sum_{t:(y_t=u)} \sum_j \gamma_{ij}(t)}{\sum_t \sum_j \gamma_{ij}(t)} \tag{16}$$

3.4.4. Algorytm uczenia genetycznego

Wyżej przedstawiony algorytm uczenia HMM FORWARD-BACKARD jest skomplikowany i co najgorsze nie daje żadnych gwarancji optymalizacji globalnej. Znajduje on jedynie lokalne maksimum prawdopodobieństwa $P(y_1^T)$.

Tabela 1

Wyniki treningu genetycznego sieci neuronowej

Nazwa sieci	Wstępne wytrenowanie	Ilość przebiegów szlifujących	Ilość pokoleń	Współczynnik mutacji [%]	Współczynnik krzyżowania [%]	Początkowy błąd średniokwadratowy dla zbioru uczącego	Końcowy błąd średniokwadratowy dla zbioru uczącego	Końcowy błąd średniokwadratowy dla zbioru testowego	Końcowy błąd średni dla zbioru testowego
gen1	tak	50	6000	2	50	0.049	0.014	0.31	0.14
gen2	nie	25	12000	3	50	0.58	0.005	0.30	0.13
gen3	tak	50	6000	2	50	0.024	0.010	0.35	0.15
gen4	nie	0	20000	3	50	0.58	0.012	0.33	0.14
gen5	nie	0	20000	5	0	0.58	0.004	0.28	0.11
gen6	nie	25	12000	5	50	0.58	0.004	0.40	0.19
gen7	nie	25	12000	10	50	0.58	0.003	0.26	0.09

Dlatego wydaje się, że celowe może być zastosowanie prostych mechanizmów algorytmów genetycznych do procesu uczenia HMM. Wydaje się, że problemem jest właściwe zdefiniowanie operacji krzyżowania. Jednakże z naszych doświadczeń z algorytmami genetycznymi stosowanymi do uczenia NN realizujących rozpoznawanie kontekstowe, ale niezwiązane z przetwarzaniem mowy (tabela 1) wynika, że sama operacja mutacji, choć bardzo prosta do implementacji, jest stochastycznym procesem dającym dobre rezultaty, zbliżone do opti-

mum globalnego, o ile ewolucja przebiega przez dostatecznie długo pokoleń [15]. Zastosowanie tych mechanizmów do HNN nie powinno w niczym zmieniać ich istotnych cech, obserwowanych przy NN, jednakże jak dotąd nie przeprowadziliśmy eksperymentów potwierdzających (bądź obalających) tę hipotezę.

3.5. Sieci neuronowe w rozpoznawaniu mowy

Jak wcześniej wspomniano, alternatywą wobec HMM w rozpoznawaniu mowy jest zastosowanie architektur NN, jako doskonale radzących sobie z rozpoznawaniem kontekstowym. W procesie uczenia najpopularniejszych, wielowarstwowych sieci typu FEED-FORWARD uczonych z nauczycielem (SUPERVISED LEARNING) stosuje się algorytm backpropagation [2, 9, 12, 15]. Aby móc go stosować, trzeba dysponować następującymi danymi uczącymi:

X=<X(1), X(2), .. ,X(k)>,

Z=<Z(1), Z(2), .., Z(k)>, k-ilość kroków uczenia

gdzie:

X(i)=[$x_1, x_2, \ldots, x_n$], n-ilość neuronów warstwy wejściowej

Z(i)=[$z_1, z_2, \ldots, z_m$], m-ilość neuronów warstwy wyjściowej

Wtedy możemy wyznaczyć wektor błędu $E(i) = [e_1, e_2, \ldots, e_m]$ dla i-tego kroku uczenia jako:

$$E(i)=Z(i)-Y(i) \tag{17}$$

gdzie:

Y(i)=[$y_1, y_2, \ldots, y_m$] wektor wyjściowy generowany przez sieć w kroku i.

Oprócz tego stosowane są nieraz NN uczone bez nauczyciela (UNSUPERVISED LEARNING), dla których nie jest znany ciąg Z, a sieć sama dokonuje klasteryzacji danych wejściowych. Sieci takie mogą być zarówno typu FEED-FORWARD, jak i rekurencyjne (ze sprzężeniami zwrotnymi).

3.5.1. Prawdopodobieństwo a'posteriori

Dodatkowym bardzo istotnym argumentem za zastosowaniem NN w systemach rozpoznawania mowy było odkrycie dokonane niezależnie przez kilka zespołów badawczych na początku lat 90, że sieć uczona metodą backpropagation na wyjściach swoich aproksymuje prawdopodobieństwo a'posteriori $P(c|x)$, gdzie c jest jedną z klas, do których należy zakwalifikować wygenerowany sygnał x. Okazuje się, że maksymalizacja prawdopodobieństwa a'posteriori pozwala na lepsze rozróżnianie między klasami w przypadku skończonej ilości wzorców uczących, niż prawdopodobieństwo warunkowe $P(x|c)$, które maksymalizowane jest w algorytmie uczącym forward-backward wykorzystywanym w HMM.

3.5.2. Sieć neuronowa TDNN

Ogólnie, sieci neuronowe w systemach rozpoznawania mowy mogą mieć informację wejściową podawaną statycznie lub dynamicznie. W sieciach statycznych informacja o całej jednostce rozpoznawania, w postaci wielu ramek, podawana jest na wejścia sieci. Dlatego sieci takie mogą służyć jedynie do rozpoznawania poszczególnych fonemów lub słów, o ile są one izolowane. W rzeczywistości były z powodzeniem budowane tego typu systemy, jednak ich skalowalność na problemy rozpoznawania dużej liczby słów, poprzez zwiększanie liczby neuronów wyjściowych, jest nieelegancka, a próby przejścia na rozpoznawanie mowy ciągłej wręcz niemożliwe. Natomiast w podejściu dynamicznym sieć neuronowa widzi tylko fragment całej informacji wejściowej przez okno przesuwające się po osi czasu. W tym wypadku łatwiejsze jest przejście od systemów rozpoznających fonemy, poprzez słowa izolowane, do systemów rozpoznających mowę ciągłą.

Aby uniezależnić odpowiedzi sieci od problemów związanych ze złym dopasowaniem czasowym, zaproponowano NN z opóźnieniem czasowym, czyli TIME DELAYED NEURAL NETWORK (TDNN). W sieciach tych uzyskuje się niezależność odpowiedzi od pozycji charakterystycznych cech lokalnych sygnału w czasie, dzięki zastosowaniu tzw. połączeń ze związanymi wagami. Związane wagi muszą przyjmować tę samą wartość. Dlatego niezbędne okazało się zmodyfikowanie algorytmu wstecznej propagacji błędu, tak by wagi związane przyjmowały nowe wartości wynikające z błędu średniego na tych wagach. Na coraz wyższych poziomach dokonuje się coraz szerszego w czasie „spojrzenia" na dane wejściowe, by na ostatnim poziomie dokonać integracji lokalnych cech wykrywanych w ramkach [11].

3.6. Układy hybrydowe NN-HMM

Wyżej przedstawione NN, na polu większych problemów, nie były jednak w stanie dorównać systemom opartym na HMM, ze względu na kłopoty z modelowaniem zróżnicowania czasowego. Starano się więc dokonać syntezy obu tych podejść, tak by można było wykorzystać pozytywne właściwości każdej z nich. Jako wynik tych prac powstała specjalizowana sieć neuronowa realizująca algorytm Viterbi przedstawiona przez Lippmanna i Golda w 1987 r. Podobny charakter miało rozwiązanie podane w 1990 r. przez Bridla, tzw. AlphaNet. Była to z kolei sieć realizująca algorytm znajdowania optymalnej drogi typu FORWARD (w którym wyznaczano prawdopodobieństwo $\alpha_i(t)$ - stąd nazwa sieci). Jednakże zarówno ViterbiNet, jak i AlphaNet nie mogły służyć do rozpoznawania mowy ciągłej ani nie usuwały niedogodności związanych ze sposobem reprezentowania rozkładu prawdopodobieństw nad przestrzenią akustyczną. Jedyną ich zaletą była demonstracja możliwości użycia sieci neuronowych do realizacji określonych algorytmów charakterystycznych dla modeli Markowa oraz fakt, że algorytm ten mógł być wykonywany równolegle dla wielu słów na wielu sieciach, gdyż jedna sieć realizowała go dla jednego słowa.

Bardziej obiecujące stało się podejście, w którym zadanie wyrównania czasowego pozostawiono metodom operującym na Ukrytych Modelach Markowa, a sieci neuronowe użyte są tylko do modelowania rozkładu prawdopodobieństw w przestrzeni akustycznej. Innymi słowy, wykonują one analizę akustyczną, nie ingerując w problemy związane z różnicami czasowymi sygnału [7].

4. Generowanie mowy

W komputerowym generowaniu mowy na podstawie zapisanego tekstu, oprócz podstawowego zadania odnalezienia właściwych fonemów, pojawia się problem poprawnej intonacji. Jedną z faz tego procesu jest wydzielenie w tekście jednostek większych niż fonemy, np. sylab, słów, fraz. Z tych zadań wyszukanie słów jest zadaniem trywialnym, podczas gdy wyszukiwanie sylab czy fraz nie jest już takie proste.

4.1. Wyodrębnianie sylab

Podział na sylaby, należący do klasy zadań rozpoznawania kontekstowego, nie jest w języku polskim określony jednoznacznymi regułami i dlatego nie jest problemem banalnym. Pomijając nieeleganckie metody bazujące na potężnych słownikach, najrozsądniejsze wydaje się zastosowanie w tym celu NN. Jednakże statyczna NN, akceptująca wyrazy długie do 20 liter włącznie, również wymagałaby ogromnej ilości pamiętanej informacji, gdyż dla i=20*32 (w języku polskim są 32 litery) oraz o=20, (i oraz o oznaczają ilość neuronów warstwy wejściowej i wyjściowej odpowiednio), rozmiar warstwy ukrytej h wyniósłby szacunkowo według potwierdzonej w [12, 15, 16] reguły

$$h = \frac{1}{2}(i + o) \tag{18}$$

około 330 neuronów, a ilość połączeń między warstwą wejściową a ukrytą c_{ih}, zgodnie ze wzorem:

$$c_{ih} = (i + 1)h \tag{19}$$

wyniosłaby 211530. Na zapamiętanie tylu połączeń potrzeba ok. 800 kB, przy założeniu 32-bitowej reprezentacji pojedynczej wagi w_{ij}. Dlatego w sieci przez nas zaprojektowanej użyto opisanego wcześniej podejścia dynamicznego, gdzie nie ma ograniczenia na długość analizowanych słów, a rozmiar sieci jest mniejszy. Zaprojektowana w ten sposób NN, poddana następnie uczeniu metodą backpropagation ze strojeniem tolerancji uczącej od 0.4 do 0.1 osiągnęła współczynnik poprawności odpowiedzi 98%.

4.2. Sieć neuronowa typu NetTalk

Podstawowym zadaniem w generowaniu mowy powinno być zbudowanie NN typu NetTalk o architekturze bardzo zbliżonej do NN z poprzedniego rozdziału, odwzorowującej w zależności od kontekstu litery w kody fonemów. Kody te mogłyby stanowić indeksy tablicy z nagranymi próbkami fonemów bądź być kierowane bezpośrednio na wejście specjalizowanych układów, np. VotrxSC-01 [12] generujących odpowiednie dźwięki. Sieć taka, wykorzystująca dynamiczne podejście rozpoznawania kontekstowego (okno przesuwne), została zbudowana na potrzeby języka angielskiego. Trudnością powstrzymującą nas przed powtórzeniem tego eksperymentu, ale dla języka polskiego, jest fakt, iż nie dysponowaliśmy polskim słownikiem z transkrypcją fonetyczną występujących w nim słów. Dostępność takiego słownika ułatwiłaby w znaczący sposób stworzenie zbiorów uczących i realizację projektu.

5. Wnioski

Na podstawie zaprezentowanego materiału można wyciągnąć wniosek, że do celów rozpoznawania kontekstowego w systemach rozpoznawania mowy NN są lepszym kandydatem niż HMM. Jest tak ze względu na to, iż NN nie posiadają bardzo ograniczającego założenia, które charakteryzuje HMM, a mianowicie wymagania niezależności. Wymaganie to implikuje duże trudności z przekazaniem kontekstu do HMM (właściwie można to zrobić jedynie stosując modele akustyczne o dużej czułości na kontekst), a zatem minimalizuje korzyści, jakie system może odnieść z właściwego wykorzystania informacji o kontekście (w rozpoznawaniu kontekstowym jest to słabość bardzo poważna). Inną słabością HMM sygnalizowaną uprzednio jest przyjmowane a'priori założenie o kształcie powierzchni gęstości prawdopodobieństwa w przestrzeni akustycznej. Wielowarstwowe NN nie mają tu żadnych ograniczeń. Natomiast ze względu na to, że problem czasowego wyrównania jest lepiej rozwiązywany przez HMM (jest to praktycznie jedyna, ale istotna korzyść ze stosowania HMM), ostatnie znaczące sukcesy na polu rozpoznawania mowy ciągłej zanotowały hybrydy NN-HMM [7]. Natomiast w zadaniu generacji mowy, gdzie wejście stanowi informacja pisana, którą bardzo łatwo „szatkować" w czasie, w tego typu zadaniach NN osiągają sukcesy w pojedynkę, co zostało potwierdzone przez nas we wcześniej opisanym eksperymencie rozpoznania sylab.

LITERATURA

1. Porto V. W.: Neural-Evolutionary Systems. Handbook of Evolutionary Computation, 1997, s.D1.1:1-D1.3:2.

2. Noyes J. L.: Learning Rules. Handbook of Neural Computation, 1997, s.B3.3:1-B3.3:10.
3. Basztura Cz.: Analiza możliwości zastosowania diafonów w automatycznym rozpoznawaniu mowy ciągłej. Problemy Współczesnej Nauki, Teoria i Zastosowania, seria Informatyka, 1996, s. 70-76.
4. Grocholewski S.: Wybrane aspekty technologii języka. Problemy Współczesnej Nauki, Teoria i Zastosowania, seria Informatyka, 1996, s. 113-117.
5. Imiołczyk J., Owsianny M.: Z prac Zakładu Fonetyki Akustycznej Instytutu Podstawowych Problemów Techniki PAN, Poznań. Problemy Współczesnej Nauki, Teoria i Zastosowania, seria Informatyka, 1996, s. 118-121.
6. Wrzoskowicz A., Gubrynowicz R.: Z prac Instytutu Podstawowych Problemów Techniki PAN, Zakład Akustyki Cybernetycznej, Pracownia Akustyki Mowy. Problemy Współczesnej Nauki, Teoria i Zastosowania, seria Informatyka, 1996, s. 193-197.
7. Tebelskis J.: Speech Recognition using Neural Networks. School of Computer Science Carnegie Mellon University, Pittsburg 1995.
8. Kilen N.: Programowanie kart dźwiękowych w Turbo Pascalu. Wydawnictwo Lynx-SFT, Warszawa 1995.
9. Hertz J., Krogh A., Palmer R. G.: Wstęp do obliczeń neuronowych. WNT, Warszawa 1995.
10. Goldberg D. E.: Algorytmy genetyczne i ich zastosowania. WNT, Warszawa 1995
11. Bodenhausen U.: Automatic Structuring of Neural Networks for Spatio-Temporal Real-World Applications. Technische Universitat Karlsruhe, Karlsruhe 1994.
12. Lawrence J.: Introduction to Neural Networks - Design, Theory, and Applications. California Scientific Software Press, Nevada City 1994.
13. Tadeusiewicz R.: Sieci neuronowe. Akademicka Oficyna Wydawnicza RM, Warszawa 1993.
14. Lindsay P.H., Norman D. A.: Procesy przetwarzania informacji u człowieka. PWN, Warszawa 1984.
15. Cyran K., Letkiewicz S., Wojciechowski P., Kołoczek D.: Zastosowanie sieci neuronowej do prognozowania wyleczenia chorych z nowotworem nerek. ZN Pol. Śl. s. Informatyka z. 33, Gliwice 1997.
16. Ciemniewski Z., Letkiewicz S., Cyran K.: Connectionist Approach in Diagnosis Support Systems on the Basis of Feedforward ANN Giving Prognosis in Urology and Cardiology. In Proc. Biomedical Engineering and Medical Informatics, Gliwice 1997.

Recenzent: Dr inż. Ryszard Winiarczyk

Wpłynęło do Redakcji 11 grudnia 1997 r.

Abstract

The paper presents the problem of context dependent recognition tasks, in particular, it describes speech recognition and speech generation HMM an NN systems. In the first part of the article, we define the problem of context dependent recognition as looking for function f such, that the formula (1) is satisfied. As speech recognition and speech generation is a specific class of such problems, we concentrate on them and describe more accurately in the rest of paper. Since speech is an example of sound, we present the acoustic channels of PC compatible microcomputers. The potential strength of new MMX technology and especially the PMADDWD command (Fig. 1) is outlined, as it concerns the systems performing context dependent recognition with the use of neural networks. Then the architecture of typical speech recognizing system is provided (Fig. 2) as well as the main tasks which are in it calculated. The signal analysis is explained on the example of Fourier Transformation (Fig. 3 and 4), but other methods are also mentioned, in particular Linear Discriminant Analysis which can be applied for further reducing of data amount which has to be processed in next steps. Figure 5 gives brief comparison between different acoustic models in terms of their context sensitivity vs. granularity. HMMs are introduced together with their most common algorithms. Forward algorithm is essentially involved in formula (8), Viterbi algorithm in formula (9), while forward-backward algorithm, which is used for HMM training, can be expressed by (15) and (16). The possibility of genetic training of HMMs is suggested, as can be read from Table 1. Alternative to HMMs is NN applying. One of its major advantages is a'posteriori probability approximation. But it has also a weakness of poor temporal modeling. So besides presenting static and dynamic approaches applied to NNs, we give the description of NN-HMM hybrid, in which temporal modeling is performed by HMM and acoustic modeling by NN as it can more easily incorporate context information and more accurately map the probability distribution over acoustic space. Last part provides short introduction to our experiment of training dynamic NN for syllable units recognition in Polish language.