STUDIA INFORMATICA 2012
Volume 33 Number 2B (106)

Dominik NIEWIADOMY, Anna KOWALCZYK-NIEWIADOMY, Adam PELIKANT
Politechnika Łódzka, Instytut Mechatroniki i Systemów Informatycznych

AUTOMATYCZNE WYZWALACZE AUDIO DLA MOWY POLSKIEJ NA BAZODANOWEJ PLATFORMIE ORACLE

Streszczenie. W artykule tym zaprezentowano autorski system automatycznych wyzwalaczy audio dla bazodanowej platformy Oracle. Opisano nowatorski proces wykrywania słów kluczowych oraz pokazane zostały wskaźniki jakościowe.

Słowa kluczowe: wyzwalacze audio, detekcja słów kluczowych w mowie

AUTOMATIC AUDIO TRIGGERS FOR POLISH SPEECH AT DATABASE PLATFORM

Summary. This article presents novel automatic audio triggers system designed for Oracle database platform. The article contains novel keyword spotting process and classification experiments results.

Keywords: audio triggers, keyword spotting

1. Wprowadzenie

Badając obecny stan wiedzy w dziedzinie baz danych, nie udało się odszukać w pełni funkcjonalnego i zaimplementowanego na platformie bazodanowej mechanizmu pozwalającego na definiowanie i wykorzystywanie wyzwalaczy opartych na nagraniach mowy. Potencjalne spektrum zastosowań takich mechanizmów skłania do przeanalizowania tego zagadnienia i do próby stworzenia systemu, który choć w części realizowałby tak zdefiniowane zadanie.

Powstanie języka SQL wynikało z konieczności pobierania informacji z relacyjnych baz danych. Do implementacji mechanizmu triggerów audio skłaniać może wiele czynników. Dzięki zastosowaniu ww. mechanizmu możliwa będzie automoderacja treści audio, co przy dużych systemach znacząco ułatwi pracę operacyjną. Dodatkowo możliwe będzie wykonanie

tzw. rozwiązań pod klucz, wykorzystujących elementy wyzwalaczy audio, jako akcje uruchamiające własne funkcjonalności stworzone na konkretne potrzeby. Zaproponowany mechanizm wyzwalaczy audio rozszerza zatem funkcjonalność systemu bazodanowego.

Samo zagadnienie automatycznych wyzwalaczy audio można połączyć z zagadnieniami wyszukiwania przez nucenie, czyli tzw. Query by Humming [1]. Takie podejście mogłoby wiązać się z wyzwalaczami muzycznymi opartymi na wysokości dźwięku, porównywanie obwiedni bądź analizie kodu Parsonsa. Co jednak przyniosłoby największe korzyści, to powiązanie wyzwalaczy audio bezpośrednio z analizą mowy i systemami ASR. Podsumowując, tak sformułowane zagadnienie można ściśle powiązać z wyszukiwaniem określonych sentencji mówionych w nagraniach mowy. Wyszukiwanie słów kluczowych w mowie, czyli keyword spotting [2], jest częścią gałęzi nauki zajmującej się automatyczną klasyfikacją dźwięku. Głównym celem algorytmów keyword spottingu jest umożliwienie systemowi wyszukiwania w plikach audio wzorców, na podstawie zapytań, bez konieczności wykonywania pełnej transkrypcji mowy na tekst oraz izolacji poszczególnych słów ze strumienia dźwiękowego. Forma zapytania (wyszukiwanego słowa kluczowego) jest zależna od implementacji i może być oparta nao próbce audio, wcześniej wykonanej transkrypcji fonetycznej bądź tekście. W wyniku tak wykonanego zapytania system powinien zwrócić listę lokalizacji poszczególnych potencjalnych wystąpień szukanego słowa w nagraniu przeszukiwanym.

W niniejszym artykule autorzy pokażą własny algorytm keyword spottingu, w pełni możliwy do wykorzystania na platformie bazodanowej. W związku z faktem, iż algorytm tego typu musi cechować duża elastyczność, przyjęto w nim wymaganie niezależności od modelu językowego oraz od mówcy. W dalszej części artykułu przedstawiony zostanie proces klasyfikacji oraz zaprezentowane zostaną wyniki testów opierające się na nagraniach mowy pochodzące z referencyjnej bazy nagrań języka polskiego CORPORA [3].

Na zakończenie zaprezentowane zostaną wyniki analizy krzywych Receiver Operating Characteristic klasyfikatora odpowiedzialnego za keyword spotting, ze szczególnym uwzględnieniem pola powierzchni pod krzywymi oraz wartościami czułości i specyficzności w funkcji typu wykorzystanych współczynników opisujących ramkę sygnału (mel cepstralne MFCC oraz human cepstralne HFCC [4]).

2. Autorski system ATA

Podstawowymi celami badawczymi, realizowanymi przez autorów pracy, były projekt oraz weryfikacja nowatorskiego klasyfikatora opartego na parametrach mel cepstralne i human cepstralne, wykorzystujących nowoczesne algorytmy programowania dynamicznego SPRING DTW oraz budowę fonemowych książek kodowych. Tak postawione zagadnienie badawcze

wiąże ze sobą cel aplikacyjny, którym jest implementacja modelowego systemu zawierającego ww. klasyfikator. W związku z tym, w ramach badania właściwości klasyfikatora zaprojektowany został system automatycznych wyzwalaczy audio, dalej nazywany system ATA (Automatyczne Triggery Audio), którego ogólny zarys przedstawiony został w [5] (obecna wersja algorytmu znacząco odbiega od wersji przedstawionej w 2010 roku). System ten ma na celu praktyczną weryfikację działania autorskiego klasyfikatora słów kluczowych z wykorzystaniem bazy danych Oracle RDBMS.

System ATA jest zatem systemem automatycznej moderacji treści audio, rozszerzając funkcjonalność bazy danych o wyzwalacze audio. Poniżej przedstawione zostały podstawowe funkcjonalności systemu dostępne z punktu widzenia użytkownika końcowego:

- **automoderacja pozytywna** – automatyczne oznaczanie treści, jako zawierającej słowa znajdujące się na liście zdefiniowanej przez administratora (np. wyzwalacze oznaczające nagrania radiowe zawierające np. nazwisko, miasto, państwo itp.),
- **automoderacja negatywna** – automatyczne oznaczanie treści audio, jako treść nie zawierająca słów ze zdefiniowanej przez administratora listy (np. lista wulgaryzmów, lista słów podejrzanych),
- **automoderacja definiowana przez użytkownika** – udostępnienie API pozwalającego w dowolny sposób definiować własne akcje automatyczne przez co możliwe jest tworzenie tzw. systemów "pod klucz",
- **definicja bazy wzorców wyzwalaczy** – definiowanie listy słów kluczowych wyszukiwanych w nowo dodawanej treści. Funkcjonalność ta pozwala na dodawanie słów kluczowych do listy wyszukiwać wraz z podaniem w kontekście, którego wyzwalacza ma dane słowo obowiązywać.

Dodatkowo w/w klasyfikator zastosowany w systemie ATA powinien być zdolny do procesu dostosowywania się do różnych języków jedynie na podstawie bezkontekstowych definicji transkrypcji fonetycznych (np. słownik nagrań z określeniem punktu początku i końca fonemów, difonów, trifonów itp.). Biorąc pod uwagę poprzednie wymaganie, system ATA zbudowano bazując na zbiorze nagrań mowy CORPORA, przy czym proces uczenia klasyfikatora oparto o transkrypcje z granulacją fonemową.

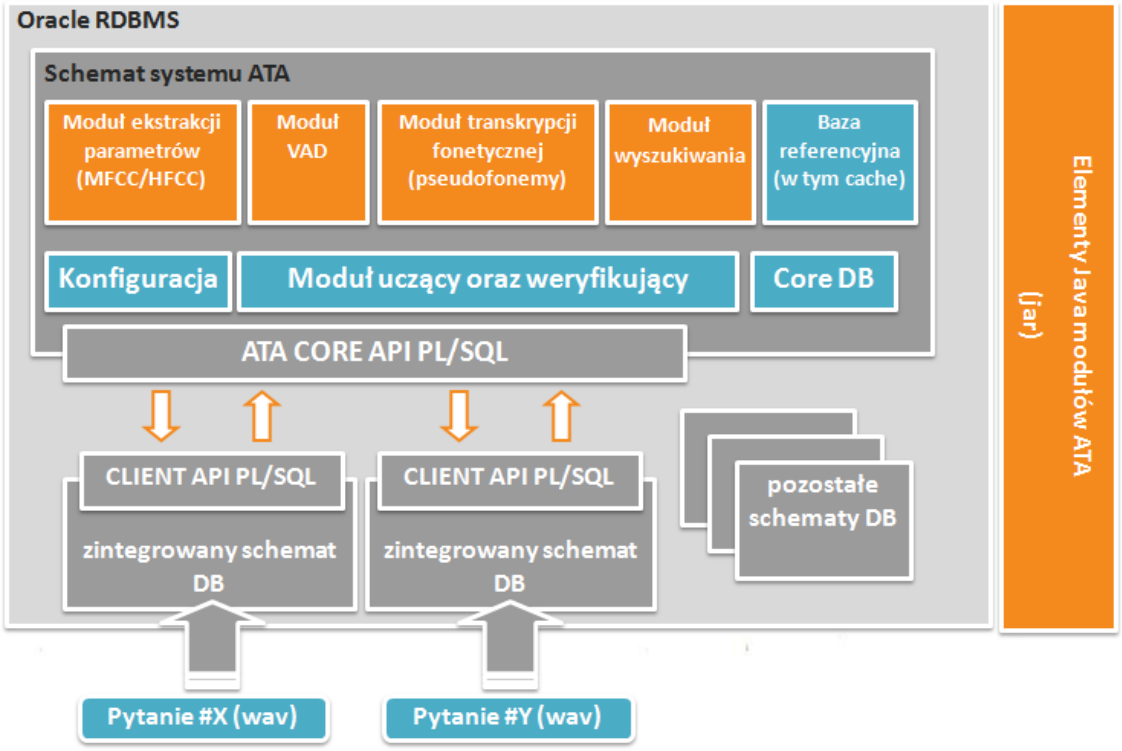
System ATA zaprojektowany został modularnie, dzięki czemu możliwa jest zmiana poszczególnych komponentów w celu jego dalszego rozwoju.

Jak widać na schemacie przedstawionym na rysunku 1 z algorytmicznego punktu widzenia wyróżniamy cztery podstawowe bloki:

- blok uczący – dostosowanie algorytmu do transkrypcji językowych na podstawie wejściowej bazy językowej (np. CORPORA dla języka polskiego, TIIMIT dla języka angielskiego, itp.);

- blok ekstrakcji parametrów – generacja parametrów MFCC/HFCC na podstawie nagrań audio (np. pliki wav, strumień próbek z mikrofonu, itp.);
- blok transkrypcji fonetycznej – konwersja serii czasowych MFCC/HFCC na serie czasowe hipotez pseudofonetycznych (pseudofonemy, gdyż możliwa jest dowolna granulacja np. oparta o fonemy, difony, trifony lub sylaby);
- blok wyszukiwania fonetycznego - wyszukiwanie algorytmem nieliniowej transformacji czasu SPRING DTW [6] podsekwencji fonetycznych z referencyjnej bazy w przeszukiwanym nagraniu. Zadanie to ma na celu weryfikację hipotezy stawianej przez wyzwalacz.

W dalszej części zaprezentowany zostanie opis procesu wyszukiwania słów kluczowych w mowie autorskim algorytmem klasyfikacji wraz z prezentacją uproszczonych bloków pseudokodu w/w modułów.



Rys. 1. Schemat ogólny systemu ATA
Fig. 1. System ATA modular design

3. Klasyfikacja występowania słów kluczowych w nagraniu mowy

Podstawowym celem prowadzonych badań była realizacja autorskiego algorytmu klasyfikującego występowanie słów kluczowych w nagraniach mowy. Tak postawione zadanie można było sprowadzić do zagadnienia implementacji klasyfikatora przyporządkowującego nagranie przeszukiwane do 2 klas dla każdego wyszukiwanego słowa kluczowego: podse-

kwencja występuje bądź podsekwencja nie występuje. Proces klasyfikacji musi zostać poprzedzony przygotowaniem konfiguracji poprzez odpowiednie dobranie książki kodowej oraz metryki fonemowej dla całego zestawu fonemów. Na wstępie dla każdego fonemu za pomocą zmodyfikowanego algorytmu klasteryzacji k-medoids dobierany jest podzbiór nagrań głosek najlepiej oddający cały zbiór nagrań w obrębie danego fonemu.

```
Jeżeli: dla danego fonemu istnieje już wcześniej wygenerowana książka kodowa
   Załaduj optymalną książkę kodową i zapamiętaj, jako OPT_KK, z optymalnej książki
   kodowej odczytaj i zapamiętaj SMAE_KK;
Inaczej:
   Brak książki kodowej OPT_KK, SMAE_KK = plus nieskończoność;
Pętla1: powtarzaj R razy:
   PP = 0;
   Pętla2: powtarzaj dopóki PP < MAX_PP
      MAE_KK = plus nieskończoność;
      prevMAE_KK = plus nieskończoność;
      Wybierz losową książkę kodową ze zbioru dostępnych fonemów i zapamiętaj ją,
      jako RAN_KK;
      Do każdego fonemu ze zbioru x, przyporządkuj indeks fonemu z RAN_KK, dla którego
      odległość FDTW jest najmniejsza;
      Pętla3: Dla każdego medoida (elementu RAN_KK) wybierz listę fonemów ze zbioru x,
      które są mu przyporządkowane w kontekście poprzedniego punktu i zapamiętaj,jako xf,a
      następnie:
         Dla każdego elementu ze zbioru xf oblicz jego sumaryczną odległość od pozostałych
         punktów z tego zbioru;
         Oblicz minimum w/w odległości i w RAN_KK zamień bieżacy medoid na ten element xf,
         który ma najmniejszą odległość od całego zbioru;
      koniec pętli 3:
      Dla książki kodowej RAN_KK zweryfikuj, w jakim stopniu dana książka kodowa oddaje
      cały zbiór x. W tym celu, dla każdego elementu zbioru x oblicz jego najlepsze
      dopasowanie (minimalna odległość) do książki kodowej i zsumuj te wartości.
      Zapamiętaj wynik operacji w MAE_KK;
      Jeżeli MAE_KK < prevMAE: PP=0 (iteracja optymalizująca);
      Jeżeli MAE_KK >= prevMAE: PP=PP+1 (iteracja jałowa);
   koniec pętli 2:
   Jeżeli MAE_KK < SMAE_KK:
   Zapamiętaj i zapisz RAN_KK jako optymalna książka kodowa OPT_KK;
   Zapamiętaj i zapisz SMAE_KK;
koniec pętli 1:
Zwróć optymalną książkę kodową OPT_KK;
```

Rys. 2. Pseudokod procesu generacji książek kodowych
Fig. 2. Codebook generation pseudo code

Rysunek 2 przedstawia pseudokod algorytmu klasteryzacji. Jako R określona jest liczba iteracji procesu dostosowywania, PP oznacza liczniki pozwalający oszacować liczbę iteracji jałowych, a MAE_KK i SMAE_KK oznaczają bieżącą i najlepszą odległość wszystkich wy-

typowanych nagrań w procesie uczenia danego fonemu od próbek wybranych do książki kodowej.

W kolejnym etapie po przygotowaniu książki kodowej budowana jest metryka fonemowa. Metryka ta definiuje krzyżową odległość pomiędzy wszystkimi fonemami. Budowana jest ona na zasadzie statystycznej odległości między nagraniami fonemów wykorzystanymi w procesie uczenia biorąc pod uwagę wszystkie możliwe kombinacje. Ponieważ podczas budowania metryki mamy do czynienia z sekwencjami ramek współczynników MFCC bądź HFCC o niejednolitej długości, matematycznie odległość obliczana jest za pomocą algorytmu szybkiej nieliniowej transformacji czasu (FDTW).

Posiadając już w pełni skonfigurowany klasyfikator, nagrania mowy (zarówno słowo kluczowe jak i element przeszukiwany) zamieniane są na serie czasowe wektorów współczynników cepstralnych. Następnie korzystając z algorytmu zaprezentowanego na schemacie 3 oraz z wcześniej zdefiniowanych książek kodowych poszczególne ramki sygnału zamieniane są do postaci najbardziej prawdopodobnych hipotez fonetycznych, zakładając potencjalne rozciągnięcie na poziomie 6 ramek (S_MIN równe -2 i S_MAX równe 4 ze skokiem S_INC równym 2).

```
Jeżeli pierwsze uruchomienie to:
   Wczytaj książki kodowe i bieżącą konfigurację;
Wygeneruj współczynniki MFCC/HFCC dla pliku nagrania mowy i zapamiętaj je, jako CC_RAW
pętla 1: dla każdej ramki serii czasowej CC_RAW oznaczonej, jako n
   pętla 2: dla każdego fonemu z książki kodowej
      Oblicz średnią długość fonemu w danej książce kodowej i zapisz, jako len
      pętla 3: dla każdego elementu bieżącej książki kodowej oznaczonego, jako EKK:
         min = plus nieskończoność;
         pętla 4: dla s = S_MIN .. S_MAX ze skokiem S_INC
            Oblicz FDTW miedzy EKK a wycinkiem CC_RAW(n,n+len+s) zapamiętaj w dist;
            Jeżeli dist < min: min = dist;
         koniec pętli 4:
         Zapamiętaj wartość min dla EKK w zbiorze EKK_SS;
      koniec pętli 3:
   koniec pętli 2:
   Posortuj EK_SS rosnąco pod kątem wartości dist i zachowaj tak przygotowany wektor
   przypisując go do n-tej ramki w CC_HYPO;
koniec pętli 1:
Zwróć CC_HYPO, czyli serię czasową najbardziej prawdopodobnych hipotez fonetycznych;
```

Rys. 3. Pseudokod procesu generowania hipotez fonetycznych
Fig. 3. Phonetic hypothesis generation pseudo code

W kolejnym kroku posiadając pełną transkrypcję sygnału do postaci ciągu hipotez fonetycznych możliwe jest wykorzystanie zmodyfikowanego algorytmu SPRING DTW. Podstawowa idea wykorzystania nieliniowej transformacji czasu typu SPRING wynika z faktu, iż

mamy ponownie do czynienia z sekwencjami o niestałej długości, przy czym nie dysponujemy już możliwością wykorzystania standardowej metryki np. Euklidesowej. Dodatkowo w obrębie jednego nagrania słowo kluczowe może pojawić się wielokrotnie, zatem modyfikacja i stosowanie klasycznego algorytmu DTW skutkowałoby zbyt dużą złożonością obliczeniową. Autorska modyfikacja algorytmu SPRING DTW (rysunek 4) bazuje, zatem na zamianie metryki numerycznej na metrykę fonemową oraz umożliwienie wykrywania wielu podsekwencji hipotez w obrębie pojedynczego pliku. Parametrem sterującym progiem wykrywania jest parametr ε, dla którego wartości przedstawione zostaną w części eksperymentalnej. Poniżej zaprezentowany algorytm bazuje na lokalnym ograniczeniu typu I algorytmu SPRING DTW:

```
Ustaw warunki brzegowe (d,dPrev, s, sPrev, dMin)zgodnie z alg. SPRING DTW
pętla 1: dla każdego t od 1 do N (czyli liczba ramek nagrania 1)
   pętla 2: dla każdego i od 1 do M (czyli liczba ramek nagrania 2)
      Ustaw wartości zmodyfikowanej macierzy STWM w wektorach d[i] oraz s[i] opierając
      się na wartościach wektorów X i Y, zadanej metryce (w numerycznym algorytmie
      Metryka  Euklidesowa) oraz lokalne ograniczenie typu I;
   koniec pętli 2:
   Pobierz wartość podobieństwa d[M] i zapisz w dM;
   Jeżeli (dM <= ε oraz dM < dMin):
      Ustaw potencjalny początek (ts) podekwencji na s[M];
      Ustaw potencjalny koniec (te) podsekwencji na (t+1);
      Ustaw dMin na wartość dM  ;
   Jeżeli (dmin <= ε oraz dm najmniejsze w całym wektorze d[i]):
      Zapamiętaj podsekwencję (ts, te) w WYNIK;
      Blokuj dalsze wykrywanie bieżącej podsekwencji ustawiając wartości graniczne;
   Przepisz wartości d do dPrev;
   Przepisz wartości s do sPrev;
koniec pętli 1:
Dodaj do WYNIK ostatni zakres jeśli zbiór wykrytych podsekwencji jeśli dm < dMin ;
Zwróć WYNIK;
```

Rys. 4. Pseudokod procesu generowania hipotez fonetycznych
Fig. 4. Phonetic hypothesis generation pseudo code

Podsumowując w celu określenia przynależności do odpowiedniej klasy autorski klasyfikator musi wykonać generację współczynników mel bądź human cepstralnych. Następnie niezbędna jest transkrypcja do postaci serii czasowej najbardziej prawdopodobnych fonemów dla każdej ramki sygnału. W ostatnim kroku na podstawie transkrypcji wykonywanej jest wyszukiwanie i klasyfikacja

4. Przeprowadzone eksperymenty

W celu opisu przeprowadzonych eksperymentów należy na wstępie zdefiniować zbiór uczący, zbiór testowy, proces testu oraz metodę weryfikacji jakości. W przypadku systemu ATA, jako zbiór uczący wykorzystano 4497 plików wav pochodzących od 22 mówców. Wszystkie nagrania zawierały imiona i należały do podzbioru bazy nagrań mowy polskiej CORPORA. Przyjęta została granulacja fonemowa, przy czym dla każdego fonemu zdefiniowano książkę kodową zawierającą 5 serii czasowych współczynników cepstralnych najlepiej reprezentujących dany fonem w kontekście algorytmu klastrującego k-medoids (losowa strategia doboru medoidów). Następnie na bazie nagrań fonetycznych przygotowano mapę odległości fonem-fonem w celu zdefiniowania metryki fonemowej na potrzeby zmodyfikowanego algorytmu SPRING DTW.

Jako zbiór testowy wykorzystany został podzbiór nagrań imion bazy CORPORA w którym wyróżniliśmy 2 niepokrywające się zbiory: nagrania wyszukiwane (Q) oraz nagrania przeszukiwane (S). Zbiór wyszukujący składa się z 42, a zbiór przeszukiwany z 300 nagrań polskich imion. Dla tak zdefiniowanego zestawu danych należało wybrać metodę walidacyjną. Ponieważ głównym celem niniejszej pracy była weryfikacja jakościowa autorskiego algorytmu, jako kryterium oceny poprawności wykorzystane zostały krzywe Receiver Operating Characteristic [7]. W przypadku systemu ATA automatyczny wyzwalacz audio musi podjąć decyzję o przypisaniu nagrania wejściowego do jednej z dwu klas: na liście referencyjnej, bądź poza listą. Przekładając to na zagadnienie wykrywania słów kluczowych, system musi podjąć decyzję czy nagranie A zawiera nagranie B. W celu oceny dowolnego klasyfikatora analiza ROC wprowadza cztery klasy ocenianych wyników klasyfikacji: TP - poprawnie zaklasyfikowany, TN - poprawnie niezaklasyfikowany, FP - zaklasyfikowany, mimo iż nie powinien być, FN - pominięty, mimo iż powinien być zaklasyfikowany. Bazując na liczebności wyżej wymienionych klasach wprowadza się pojęcia czułości oraz specyficzności.

Czułość klasyfikatora wyraża proporcję elementów poprawnie zaklasyfikowanych do pozostałych elementów poprawnie oznaczonych w zbiorze testowym. Czułość jest, zatem miarą zdolności klasyfikatora do rozpoznawania wyszukiwanych elementów.

$$Czulosc = \frac{TP}{TP + FN}. \tag{1}$$

Specyficzność klasyfikatora wyraża udział nagrań, które nie powinny być zaklasyfikowane i tak się stało.

$$Specyficznosc = \frac{TN}{TN + FP}. \tag{2}$$

Finalna ocena wzajemnego stosunku wyników fałszywie dodatnich do prawdziwie dodatnich może być dokonana na podstawie analizy zależności pomiędzy czułością a specyficzno-

ścią biorąc pod uwagę czułość i specyficzność dla warunków brzegowych testu. Zależność taka definiowana jest, jako krzywa ROC. Krzywa ROC stanowi całościową ocenę klasyfikatora w pełnym zakresie przeprowadzonych testów. Krzywa ta budowana jest poprzez zestawienie na osi x wartości równej (1-specyficzność), a na osi y wartości równej czułości dla bieżącego testu.

Jako miarę jakości klasyfikatora w analizie ROC wykorzystuje się pole powierzchni pod krzywą ROC czyli tzw. AUC (ang. Area Under Curve). Im większa wartość AUC tym wyższa jakość klasyfikatora. Jeśli AUC jest mniejsze bądź równe 0.5, klasyfikator taki uznaje się za niezdatny do użytku, gdyż losowy rozkład dla 2 klasowego testu daje również AUC równe 0.5. W praktyce przyjmuje się, że wartości powyżej 0.8 oznaczają dobry poziom jakości klasyfikacji, a wartości powyżej 0.9 dają bardzo dobry poziom jakości.

Znając metodykę weryfikacji eksperymentów należy przedstawić sam proces testowy. Aby pokryć pełne spektrum krzywej ROC należy wykonać testy w funkcji współczynnika epsilon algorytmu SPRING DTW. Dla zbiorów Q i S pojedyncza iteracja składa się z 12600 operacji klasyfikacji. Na pojedynczą krzywą ROC składa się, co najmniej 54 iteracji. Dla w/w. zbiorów testowych zdefiniowany został proces weryfikacyjny opisany pseudokodem zaprezentowanym na rysunku 5:

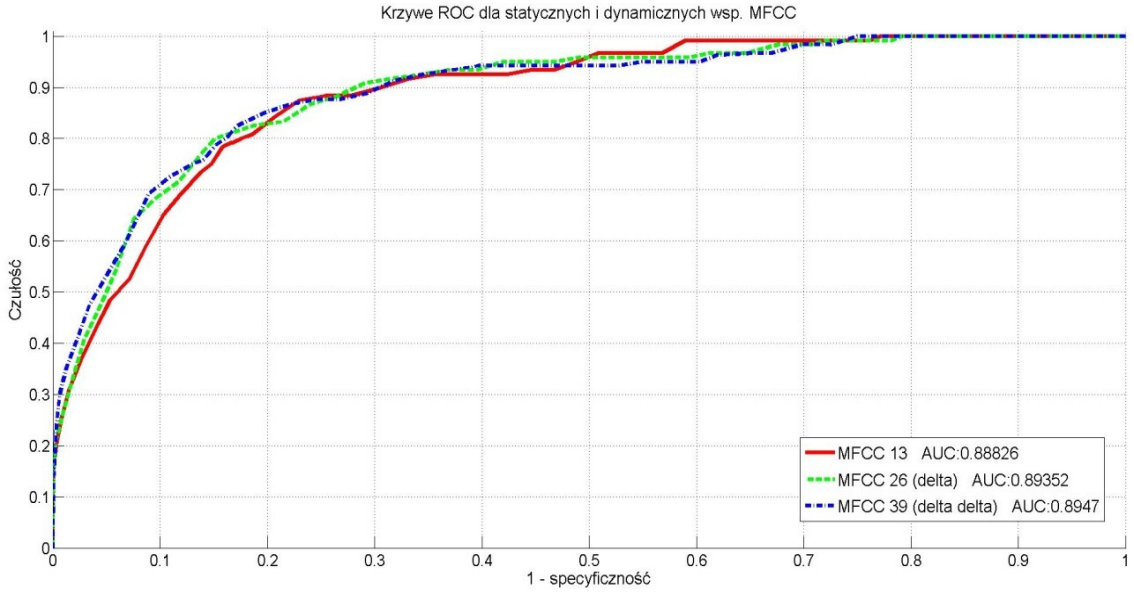
```
pętla 1: dla każdego elementu zbioru wyszukującego
   wygeneruj transkrypcję fonetyczną nagrania i zapamiętaj w cache
koniec pętli 1
pętla 2: dla każdego elementu zbioru przeszukiwanego
   wygeneruj transkrypcję fonetyczną nagrania i zapamiętaj w cache
koniec pętli 2
pętla 3: dla zadanego zakresu epsilon [0:0.15:8]
   pętla 4: dla każdego (Q) nagrania wyszukiwanego
      pętla 5: dla każdego (S) nagrania przeszukiwanego
         Wyszukaj Q w S skoryguj wartości TP/TN/FP/FN
      koniec pętli 5
   koniec pętli 4
   Zapamiętaj TP/TN/FP/FN, specyficzność, czułość, ACC dla zadanego epsilon
   Wyzeruj TP/TN/PF/FN
koniec pętli 3
Zapisz rekordy w funkcji epsilon dla danej konfiguracji
```

Rys. 5. Pseudokod procesu testowego
Fig. 5. Test process pseudo code

5. Wyniki eksperymentów

Poniżej przedstawione zostały wyniki osiągnięte przy wykorzystaniu autorskiego algorytmu klasyfikatora keyword spottingu dla 6 konfiguracji zgodnie z procesem testowym opisanym w poprzednim punkcie.

Na wstępie zaprezentowano 3 krzywe ROC dla klasyfikatora opartego o współczynniki MFCC zdefiniowane zgodnie z poniższą konfiguracją: 13 współczynników statycznych + 13 współczynników ΔMFCC + 13 ΔΔMFCC, dla ramki o długości 256 próbek z przesunięciem 100 próbek i splotem z oknem Hamminga. Jako bank filtrów wykorzystano 40 filtrów MFCC rozpiętych między 130Hz a 6800Hz.



Rys. 6. Krzywe ROC dla statycznych i dynamicznych MFCC
Fig. 6. ROC curves for static and dynamic MFCC

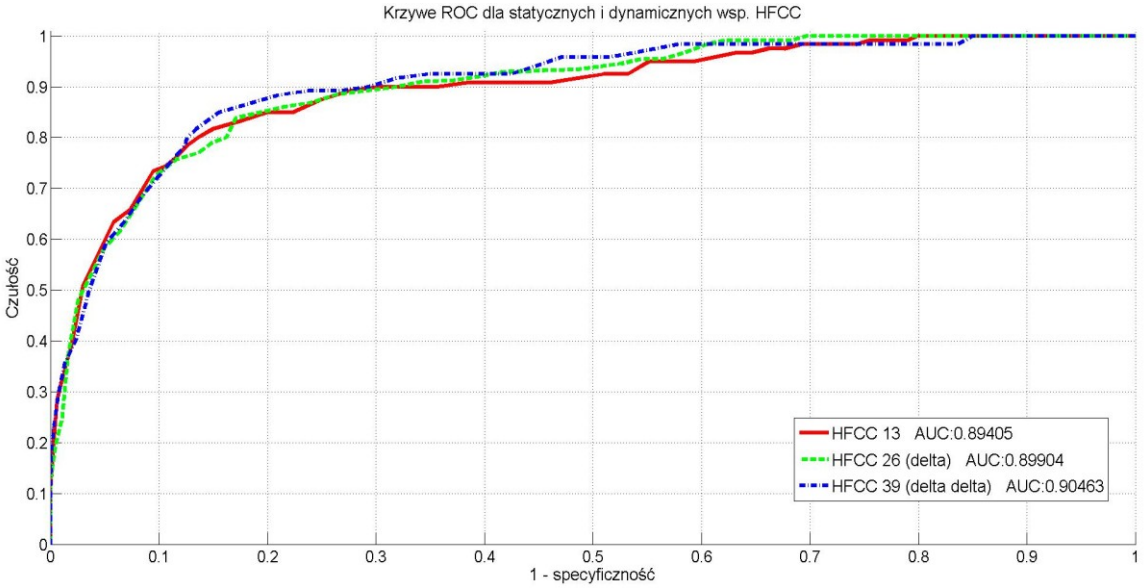
W tabeli 1 zaprezentowano zestawienie wyników dla tak zdefiniowanej konfiguracji. Jak widać najwyższe AUC dla współczynników MFCC wynosi 0.895 dla ramki 36 elementowej:

Tabela 1
Wyniki dla statycznych i dynamicznych MFCC

MFCC	AUC	Czułość	Specyficzność
13	0.888	0.8	0.824
13+13Δ	0.894	0.8	0.839
13+13Δ+13ΔΔ	0.895	0.8	0.849

Przy testowym progu skuteczności równym 80% specyficzność testowanego algorytmu wyniosła 84,9%. Pozwala to stwierdzić, iż algorytm poprawnie sprawdza się do klasyfikacji rzadkich wystąpień słów kluczowych w nagraniach mowy przy wykorzystaniu współczynników MFCC.

Kolejne 3 krzywe ROC zaprezentowane zostały dla klasyfikatora opartego o współczynniki HFCC zdefiniowane zgodnie z poniższą konfiguracją: 13 współczynników statycznych HFCC + 13 współczynników ΔHFCC + 13 ΔΔHFCC, dla ramki o długości 256 próbek z przesunięciem 100 próbek i splotem z oknem Hamminga. Jako bank filtrów wykorzystano 40 filtrów MFCC rozpiętych między 130Hz a 6800Hz o jednostkowej wysokości i współczynniku eFactor równym 1.



Rys. 7. Krzywe ROC dla statycznych i dynamicznych HFCC
Fig. 7. ROC curves for static and dynamic HFCC

W tabeli 2 zaprezentowano zestawienie wyników dla tak zdefiniowanej konfiguracji. Jak widać najwyższe AUC dla współczynników HFCC wynosi 0.905 dla ramki 36 elementowej:

Tabela 2

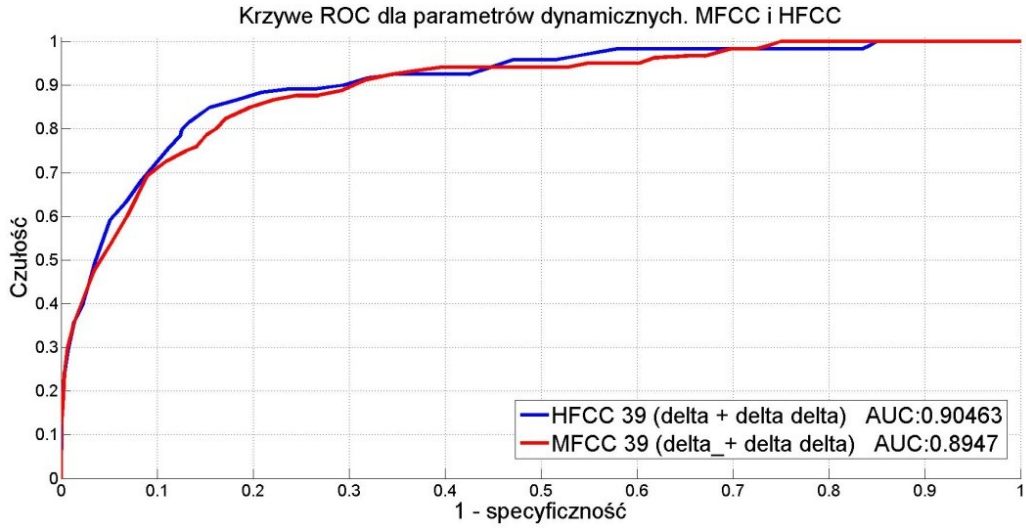
Wyniki dla statycznych i dynamicznych HFCC

HFCC	AUC	Czułość	Specyficzność
13	0.894	0.8	0.864
13+13Δ	0.899	0.8	0.868
13+13Δ+13ΔΔ	**0.905**	0.8	**0.874**

Przy testowym progu skuteczności równym 80% specyficzność testowanego algorytmu wyniosła 87,4%. Pozwala to stwierdzić, iż algorytm poprawnie sprawdza się do klasyfikacji rzadkich wystąpień słów kluczowych w nagraniach mowy przy wykorzystaniu współczynników HFCC, co było elementem analizy.

Oprócz powyższej analizy jakościowej wykonana została także analiza porównawcza mająca na celu weryfikację, który typ współczynników MFCC, czy HFCC uzyskuje lepsze wyniki w badanym zagadnieniu. Poniżej przedstawione zostały krzywe ROC dla 39 elementowych wektorów MFCC i HFCC.

Jak widać na rysunku 6 oraz bezpośrednio na danych liczbowych przedstawionych w powyższych tabelach wykorzystanie współczynników human cepstralnych pozytywnie wpływa na poprawę jakości klasyfikacji mowy. Krzywa ROC dla HFCC w niemalże całym zakresie jest ponad krzywą dla współczynników MFCC.



Rys. 8. Krzywe ROC dla statycznych i dynamicznych HFCC
Fig. 8. ROC curves for static and dynamic HFCC

6. Podsumowanie

Autorom pracy udało się zaprojektować i zaimplementować w pełni funkcjonalny i kompletny algorytm keyword spottingu pozwalający na wyszukiwanie słów kluczowych w nagraniach mowy opierając sie na dwóch rodzajach współczynników: powszechnie stosowanych współczynnikach mel cepstralnych (MFCC czyli Mel Frequency Cepstral Coefficients) oraz nowatorskich współczynnikach human cepstralnych (HFCC czyli Human Factor Cepstral Coefficients). Wykonana została seria testów w celu doboru konfiguracji, która pozwoliłaby na osiągnięcie zadawalających wyników. W ramach testów wykonano także analizę porównawczą, która pozwoliła na wyciągnięcie wniosku, iż współczynniki HFCC lepiej oddają sposób postrzegania mowy niż współczynniki MFCC, przez co podnoszą skuteczność klasyfikacji.

Sam algorytm klasyfikatora zaprojektowany został tak, aby mógł klasyfikować opierając się na nagraniach mowy w formie plików wav, jak i bezpośrednio w oparciu o transkrypcje fonetyczne.

Dodatkowo algorytm jest w pełni niezależny od języka (modelu językowego i słownika) oraz mówcy (nagrania różnych osób zarówno po stronie nagrań przeszukujących jak i przeszukiwanych). Jedyne, co potrzeba do procesu uczenia klasyfikatora to baza nagrań ze zdefiniowanymi transkrypcjami do zadanej jednostki np. fonemu. Wszystkie wcześniej wymie-

nione cechy pozwoliły na implementację w/w algorytmu na platformie serwera baz danych Oracle i zastosowanie go na potrzeby systemu Automatycznych Triggerów Audio. Implementacja algorytmu przy pomocy języka Java biorąc pod uwagę obostrzenia wynikające z docelowego położenia na platformie bazodanowej pozwalają przy pomocy procedur importu przenieść klasyfikator bezpośrednio na serwer Oracle.

Podsumowując założone cele naukowe oraz projektowe zostały osiągnięte. W pełni funkcjonalna wersja algorytmu skonfigurowanego do mowy polskiej w oparciu o bazę CORPORA pozwoliła na implementację funkcjonalności automatycznych triggerów audio.

Praca naukowa finansowana ze środków na naukę w latach 2010-2011 jako projekt badawczy

BIBLIOGRAFIA

1. Weinstein E.: Query By Humming: A Survey. MIT oraz Google, 2005.
2. Timofte R., Hautamäki V., Fränti P.: Speaker, Vocabulary and Context Independent Word Spotting System for Continuous Speech. Proc of. International Symposium on Chinese Spoken Language Processing 2006, Singapore 2006.
3. Grocholewski S.: Baza nagrań sygnałów mowy CORPORA. Instrukcja uzytkowania, Politechnika Poznańska, Instytut Informatyki, Poznań 1997.
4. Skowronski M.: Bilogically inspired noise-robus speech recognition for both man and machine. University of Florida, Florida, praca doktorska 2004.
5. Niewiadomy D., Pelikant A., Kubiak R.: Automatyczne znakowanie danych audio na platformie serwera baz danych Oracle. Studia Informatica, Vol. 31, No. 2A(89), Wydawnictwo Politechniki Śląskiej, Gliwice 2010.
6. Sakurai Y., Faloutsos C., Yamamuro M.: Stream Monitoring under the TimeWarping Distance. IEEE 23rd International Conference on Data Engineering, 2007. ICDE 2007, NTT Cyber Space Labs, 2007.
7. Lasko T. A., Bhagwat J. G., Zou K. H., Ohno-Machado L.: Methodological Review: The use of receiver operating characteristic curves in biomedical informatics. Journal of Biomedical Informatics, Vol. 38, No. 5, 2005, s. 404÷415.

Wpłynęło do Redakcji 8 stycznia 2012 r.

Abstract

This chapter presents the research done at Institute of Mechatronics And Computer Science of Technical University of Lodz in order to create Automatic Audio Triggers System (in Polish System ATA). The article presents three aspects of contribution.

The main aspect is connected to design of a novel keyword spotting algorithm which can be easily applied for any language without knowledge of its structure. The only one designed limitation for the algorithm is to have a phonetic transcription for set of audio files of that language. The proposed algorithms are based on new approach to Dynamic Time Warping SPRING algorithm which was originally presented in [6]. The Second area of this work is an evaluation of robustness of MFCC (Mel Frequency Cepstral Coefficients) and HFCC (Human Factor Cepstral Coefficients) [4]. The final comparison is presented with verdict which coefficients give better results. The last part of the chapter presents implementation results of automatic audio triggers on Oracle database platform with general design schema of ATA system.

All of those areas are presented in this chapter with experimental evaluation and the Receiver Operator Characteristic charts. The AUC (Area Under Curve) metric for some experiments is presented with proper conclusions.

Adresy

Dominik NIEWIADOMY: Politechnika Łódzka, Instytut Mechatroniki i Systemów Informatycznych, ul. Stefanowskiego 18/22, 90-924 Łódź, Polska, dominik.niewiadomy@gmail.com.

Anna KOWALCZYK-NIEWIADOMY: Politechnika Łódzka: Politechnika Łódzka, Instytut Mechatroniki i Systemów Informatycznych, ul. Stefanowskiego 18/22, 90-924 Łódź, Polska, anna.kowalczykniewiadomy@gmail.com.

Adam PELIKANT: Politechnika Łódzka: Politechnika Łódzka, Instytut Mechatroniki i Systemów Informatycznych, ul. Stefanowskiego 18/22, 90-924 Łódź, Polska, apelikan@p.lodz.pl.