EÖTVÖS LORÁND TUDOMÁNYEGYETEM
Informatikai Kar

Hangegér

hang és beszédfelismerés

Diplomamunka

Gerőfi Balázs

ELTE IK Információs Rendszerek Tanszék

témavezetők:
dr. habil. Lőrincz András
tudományos főmunkatárs
Hévizi György
tudományos segédmunkatárs

Budapest, 2005

Tartalomjegyzék

1. Bevezetés **1**
1.1. A dolgozat felépítése 1
1.2. Köszönetnyilvánítás 2

2. Analóg és digitális audió **3**
2.1. Analóg audió 3
2.2. Digitális audió 4
2.2.1. Mintavételezés 5
2.2.2. Kvantálás 7

3. Beszéd és hangfelismerésről általában **8**
3.1. Célkitűzés 8
3.2. Felépítés 9

4. Standard előfeldolgozók (Front-end) **11**
4.1. Általános áttekintés 11
4.2. Spektrum Analízis (FFT) 12
4.3. Ablak függvények (window functions) 14
4.4. Spektrum transzformációk (warping) 15
4.4.1. Bark Scale, Critcal Band Rate 15
4.4.2. Mel Scale Filter Bank (MSFB) 15
4.5. Kepsztrum 17
4.6. Mel Scale Cepstral Coefficients (MFCC) 20
4.7. Lineáris Predikciós Együtthatók (LPC) 20
4.8. Egyéb spektrális módszerek 22
4.9. Első és másodrendű delta együtthatók 22
4.10.Vektor kvantálás (VQ) 23

5. Akusztikus modellek és algoritmusaik **25**
5.1. Áttekintés, Bayes-hálók 25
5.2. Rejtett Markov Modellek 27
5.2.1. Diszkrét Markov folyamatok 27
5.2.2. Rejtett Markov Modellek 29
5.2.3. A HMM három alapproblémája 31

5.2.4. A HMM három alapproblémájának megoldásai . 31
Az 1. probléma megoldása 32
A 2. probléma megoldása, Viterbi algoritmus . . 34
A 3. probléma megoldása, Baum-Welch algoritmus 36
5.2.5. Kiterjesztés folytonos esetre 38
5.3. Alternatív modellek 40
5.3.1. Állapot fejlődés folyamat 41
5.3.2. Megfigyelés hozzárendelés 42
5.4. Lineáris Dinamikus Rendszerek (LDS) 42

6. Hang-felismerés képfeldolgozással **44**
6.1. Viola-Jones objektum detektálás 44
6.1.1. Jellemzők (features) 46
6.1.2. Integrális kép 47
6.1.3. Osztályozó függvények tanítása, Adaboost 50
6.1.4. Osztályozók kaszkádba szervezése 52
Kaszkád szervezés algoritmusa 53
6.2. Adatbázis 56
6.3. Tanítás és eredmények 56

1. fejezet

Bevezetés

A számítógépek rohamos terjedése és egyre szélesebb körü alkalmazása hatékony humán-komputer interakciós (HCI) eszközöket igényel. A humán-komputer interakció egy meghatározó szelete az emberi hang és a beszéd feldolgozásának kérdése. Kifinomult felismerő rendszerek fejlesztése a testi fogyatékkal rendelkező emberek számítógéppel történő interakcióját is jelentősen elősegítheti.

Az ELTE NIPG csoport egy része alternatív humán-komputer interakciós eszközök fejlesztésével foglalkozik testi fogyatékos gyermekek számítogéppel való kommunikációjának segítésére. Az eszközök az AAK[1] kommunikációs központban kerülnek tesztelésre. Ezen kutatási program részeként felmerült az igény egy a hangfelismerés alapján működő eszköz fejlesztésére. Az eszköz neve "Hangegér" és célja testi fogyatékkal rendelkező gyermekek számára a számítógéppel való kommunikációban az egérfunkciók kiváltása és így a sikeresebb kommunikáció lehetőségének megteremtése.

A dolgozatban ismertetjük a hang- és beszéd-felismerő rendszerek általános felépítését és működését. Alternatív megoldásként bemutatunk egy a képfeldolgozásban használt technológia alapján működő minta-felismerő rendszert.

1.1. A dolgozat felépítése

A dolgozatban először az analóg és digitális audióval foglalkozunk (2. fejezet). Megvizsgáljuk a minta-kereső és beszéd-felismerő rendszerek általános felépítését és működését (3. fejezet). Részletesen áttekintjük a felismerő rendszerek fontosabb alkotó elemeit.

[1]Augmentatív Alternatív Kommunikáció (AAK) Segítő Módszertani Központ, H-1112 Budapest XI., Neszmélyi út 36., http://www.c3.hu/ drahu/bliss/segito.html

Először az előfeldolgozókat (4. fejezet), majd az akusztikus modelleket és algoritmusaikat (5. fejezet). Végül bemutatunk egy képfeldolgozási technológiát és annak alkalmazását a hangmintafelismerés területén (6. fejezet).

1.2. Köszönetnyilvánítás

A dolgozatot az ELTE Információs Rendszerek Tanszékén működő NIPG csoport segítségével írtam. Köszönöm témavezetőimnek, Lőrincz Andrásnak és Hévizi Györgynek a gondos útmutatást és a dolgozat megírásához szükséges feltételek biztosítását. Köszönöm Szendrő Balázsnak és Kiszlinger Melindának, hogy segítségükkel hozzájárultak a dolgozat elkészítéséhez.

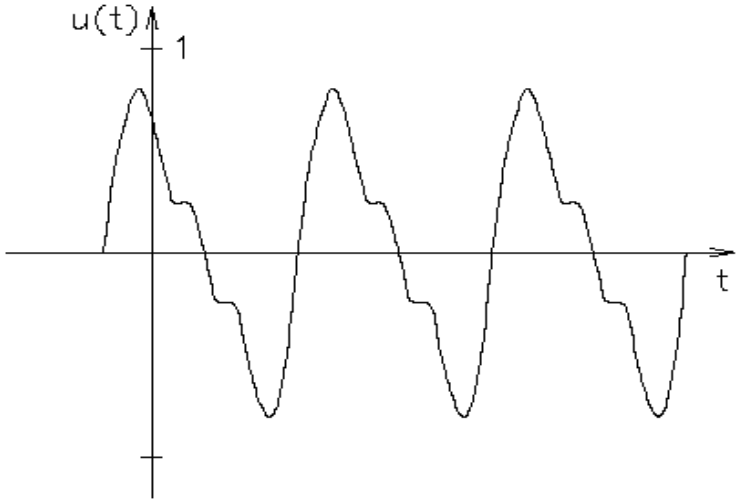
2. fejezet

Analóg és digitális audió

Mielőtt a hang- és beszédfelismerés folyamatának részleteibe mennénk tisztáznunk kell mi is az a hang és hogyan reprezentálja a számítógép a hangot. Megvizsgáljuk annak tulajdonságai és hogy hogyan is történik maga a ”hallás” a számítástechnika aspektusából.

2.1. Analóg audió

A hang, vagy technikai megfogalmazásban az audió jel a légnyomás időbeli változása. A hang analóg jel.



2.1. ábra. **Analóg jel szemléltetése folytonos függvénnyel.**

A hang elektronikus feldolgozásához konverzióra van szükség,

ami a légnyomás változását valamilyen elektronikus jelre képezi. A digitális jellel ellentétben az analóg jel tulajdonságai a következők:

- Minden időpillanatban definiált.
- Végtelen sok különböző értéke lehet.

Az analóg jelek szemléltetésének legelterjedtebb módja egy folytonos függvénnyel való ábrázolás, ezt szemlélteti a 2.1. ábra. A tény, hogy a függvény folytonos garantálja a tulajdonságokat, azaz hogy a jel minden időpillanatban definiált és végtelen sok értéke lehet.

2.2. Digitális audió

Szemben a folytonos ábrázolással digitális esetben egy tetszőleges érték leírásához véges sok szimbólum áll rendelkezésre. A latin "digit" szó azt jelenti "ujj", vagyis a legtermészetesebb szimbólumkészlet a számábrázoláshoz. A bináris digitális technika csak két két számjegyet használ, a 0-t és az 1-et. Ilyen értelemben beszélhetünk bináris digitális jelekről. Több egymás melletti bitet egy kettes számrendszer beli értékként értelmezve tetszőleges értékig "számolhatunk", de a variációk száma véges marad és két egymást követő szám közötti értékek ebben a rendszerben nem ábrázolhatók. Ezért azt mondjuk, hogy a digitális rendszer diszkrét. Egy elektronikus jel esetén mindig szükséges bizonyos idő, hogy a jel egy adott értékről egy másikra változzon, így a jel értékei csak bizonyos időközönként ábrázolhatóak. Ezért azt mondjuk, hogy a digitális rendszer időben diszkrét. A digitális jel jellemzői tehát:

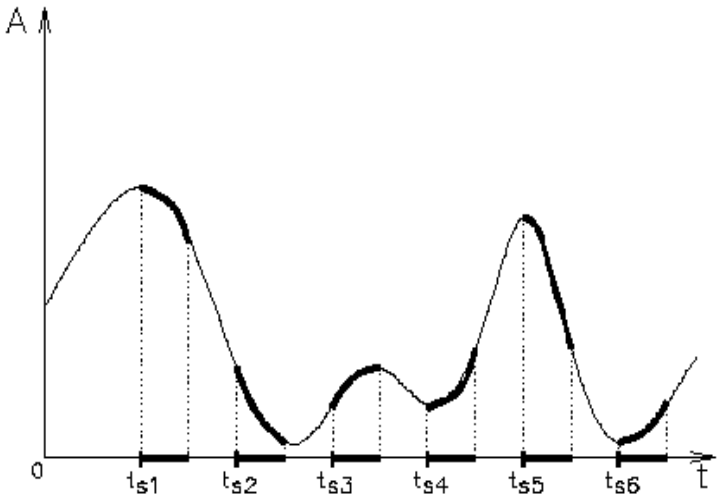
- Csak bizonyos időpillanatokban definiált.
- Véges sok különböző értéke lehet.

Amikor egy analóg jelet digitálissá alakítunk két fontos lépés szükséges:

1. Az analóg jel aktuális értékét bizonyos időnként le kell olvasni. Ezt nevezzük mintavételezésnek (Sampling).
2. A kapott értéket digitálisan, számokkal kell ábrázolni. Ezt kvantálásnak hívjuk (Quantization).

2.2.1. Mintavételezés

Ahhoz tehát, hogy egy analóg jelet digitálissá alakítsunk mintákat kell vennünk az analóg jelből. Elméletileg az egyes minták leolvasása közti időközök teljesen tetszőlegesek lehetnek, de azért, hogy az analógból digitális és a digitálisból analóggá alakítás szinkronban álljanak ezen időközöket konstansnak választják. Ebből adódik a digitális jelek egyik központi fogalma a mintavételezési frekvencia. Egy adott minta leolvasását, vagyis a minta vételezését nagyon rövid idő alatt kell megtenni, hogy az ezen idő alatt a jelben bekövetkező változás minimális legyen. Ezt szemlélteti a 2.2. ábra.

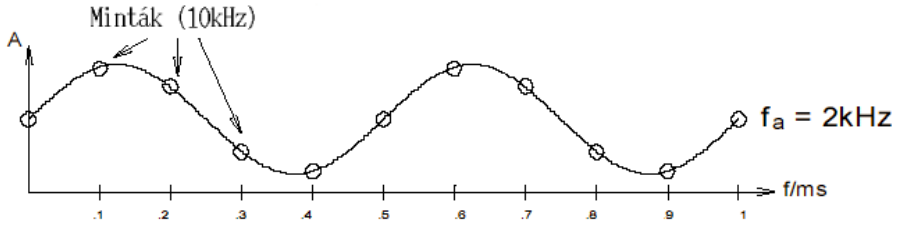


2.2. ábra. **Mintavételezés.**

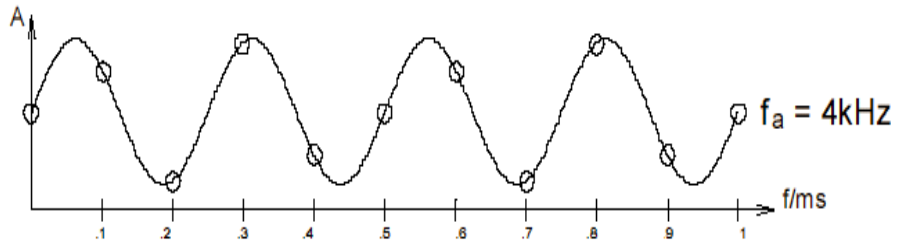
A t tengely sötét intervallumai az egyes minták leolvasásának idejét szemléltetik, az intervallumok száma a mintavételezési frekvenciát.

Mi alapján válasszuk meg a mintavételezési frekvenciát? A következőkben egy érdekes jelenséget mutatunk be. Tegyük fel, hogy a mintavételezési frekvencia most 10kHz, vagyis 10 ezer minta másodpercenként. Három esetet vizsgálunk meg, 2kHz, 4kHz és 6kHz frekvenciájú szinusz jeleket mintavételezünk.

A 2kHz-es szinusz jel mintavételezése látható a 2.3. ábrán, míg a 4kHz-est a 2.4. szemlélteti.

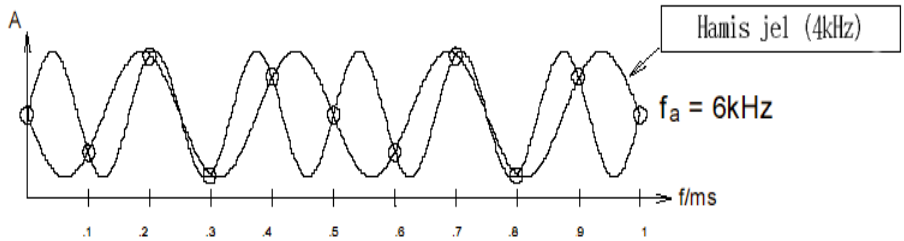


2.3. ábra. **2kHz-es jel 10kHz-es mintavételezése.**



2.4. ábra. **4kHz-es jel 10kHz-es mintavételezése.**

A 6kHz-es jel mintavételezése látható a 2.5. ábrán. A jel ilyen mintavételezés mellett nem visszaállítható, illetve a visszaállított jel hamis lesz. Ezen jelenség pontos matematikai hátteréről további információ az [24]-ben. A jelenség okáról, a mintavételezési frekvencia és a visszaállítható maximális frekvenciájú jel közti összefüggésről szól a következő tétel, melynek két elterjedt megfogalmazását ismertetjük.



2.5. ábra. **6kHz-es jel 10kHz-es mintavételezése.**

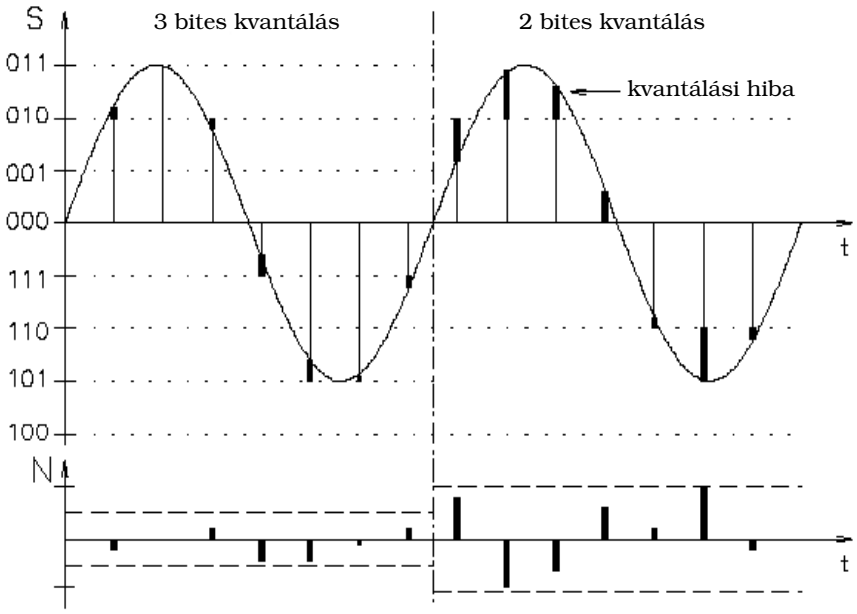
A mintavételezett analóg jel visszaállításához teljesülnie kell, hogy

- **Nyquist kritérium:** az analóg jel frekvenciája kisebb, mint a mintavételezési frekvencia fele.
- **Shannon tétel:** a mintavételezési frekvencia legalább kétszerese az analóg jel frekvenciájának.

Mindkét állítás ugyanazt a megszorítást teszi, csupán a megfogalmazás más. De miért is fontos ez számunkra a beszéd-, hangfelismerés során? Az emberi hallás 20Hz és 20kHz közötti frekvenciatartományba esik, az emberi beszéd pedig 100Hz és 8kHz közé. Ezen tényekből és az előző kritériumból adódik a CD lemez 44.1kHz-es mintavételezése és hogy miért használnak általában 16kHz-en mintavételezett hangokat a beszédfelismerés során.

2.2.2. Kvantálás

A mintavételezés után az analóg jel aktuális értékét digitálisan kell reprezentálni. Ezt a folyamatot hívjuk kvantálásnak. A kvantálás legfontosabb tulajdonsága, hogy a egyes leolvasott értékeket hány biten ábrázolja. Minél több bitet használ a számábrázolásra, annál kisebb lesz az eredeti és a digitális jel adott pillanatban vett hibája. Ezt szemlélteti a 2.6. ábra.



2.6. ábra. **Kvantálás.**

A bal oldali ábrán 3 bites, míg a jobb oldalin 2 bites kvantálás látható. Az alsó ábra az eredeti jel és a digitális érték közti hibát szemlélteti, figyeljük meg mennyivel pontosabb a bal oldali, 3 bites kvantálás.

3. fejezet

Beszéd és hangfelismerésről általában

Ebben a fejezetben ismertetjük a beszédfelismerő rendszerek általános felépítését. A következő öt alapvető építőelemre bontható egy ilyen rendszer: előfeldolgozó (front-end), másnéven jellemző kiválasztó (feature extraction), akusztikus modellek, nyelvi modell, a lexikon és a felismerő algoritmus. Ezen blokkok közül részletesen az előfeldolgozókat, az akusztikus modelleket és a felismerő algoritmust vizsgáljuk a következő fejezetekben. A többit említés szintjén az általános összképen belül tárgyaljuk.

3.1. Célkitűzés

A felismerő rendszer feladata, hogy a digitális audió jelből meghatározza mely szavak, hangok kerültek kimondásra. A továbbiakban ha azt mondjuk szósorozat, akkor az alatt nem csupán értelmes szavak, hanem bármilyen a felismerni kívánt hangot vagy azok egy sorozatát értjük. Tisztán az audió jelet hasonlítani nem célravezető, ezért a rendszer először az audió jelből kivonja annak jellemző tulajdonságait, ezt nevezik feature kiválasztásnak és ez a feladata az előfeldolgozónak (front-end). A jellemző tulajdonságok vektor formájában jelennek meg, tehát a feldolgozandó input ezután vektorok sorozata. Ezt nevezzük megfigyelés sorozatnak

$$O = \{ o_1, o_2, \ldots, o_T \}.$$

Ezek alapján a rendszer dolga a legvalószínűbb szósorozat

$$W = \{ w_1, w_2, \ldots, w_L \}$$

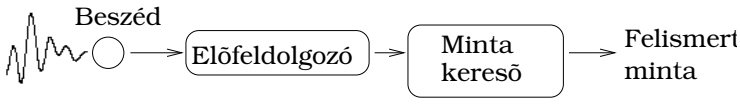
megállapítása, amelyre teljesül, hogy

$$\tilde{W} = \arg\max_{W} P(W|O), \tag{3.1}$$

vagyis amelyik szósorozat a legvalószínűbb ezen megfigyeléssorozat mellett.

3.2. Felépítés

Amennyiben a $P(W|O)$ valószínűséget közvetlenül a felismerhető szavak vagy hangokhoz rendelt megfigyeléssozorattal való összehasonlítás alapján számítjuk, akkor egyszerű mintafelismerésről beszélünk (pattern matching). Az ilyen rendszer szerkezetét szemlélteti a 3.1. ábra.

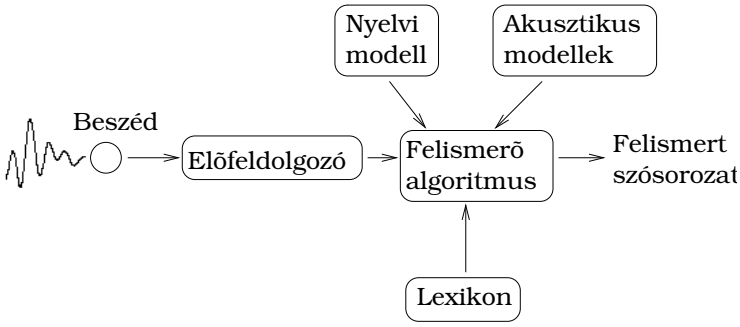


3.1. ábra. **Egyszerű minta kereső rendszer felépítése.**

Az 3.1. egyenlőségre a Bayes formulát alkalmazva kapjuk, hogy:

$$\tilde{W} = \arg\max_{W} P(W|O) = \arg\max_{W} \frac{P(O|W)P(W)}{P(O)}, \tag{3.2}$$

A képletben a $P(O)$ értéket egyszerűen elhagyhatjuk, mivel az független a szósorozattól.



3.2. ábra. **A beszédfelismerő rendszer felépítése.**

Az olyan rendszereket, melyek a maximumkeresést a jobb oldali összefüggés alapján oldják meg, beszédfelismerő rendszereknek nevezzük (speech recogniser). A felismerő rendszer tehát lényegében

a 3.2. egyenletet megoldó rendszer. Az egyes építőelemek viszonyát szemlélteti a 3.2. ábra. Az akusztikus modell feladata a $P(O|W)$ valószínűség számítása, míg a nyelvi modell a $P(W)$ érték meghatározásáért felelős.

A felismerő algoritmus nem más mint a maximum meghatározása a 3.2. egyenletben. Az algoritmus erősen kapcsolódik a használt akusztikus modellhez. A későbbiekben részletesen tárgyalt Rejtett Markov Modellek a jelenleg legszélesebb körben alkalmazott akusztikus modellek. Ebben az esetben a felismerő algoritmus a Viterbi módszer. A lehetséges megoldásra tovább korlátozást tesz a lexikonban felsorolt véges sok kombináció.

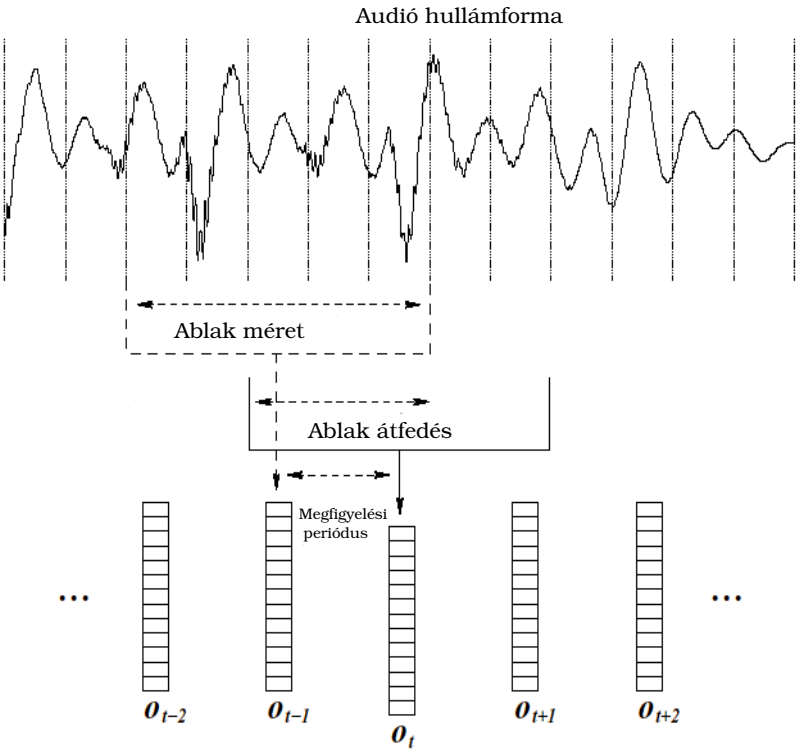
4. fejezet

Standard előfeldolgozók (Front-end)

Ebben a fejezetben az előfeldolgozót vizsgáljuk. Az előfeldolgozó feladata a beszéd jellemző tulajdonságainak kiválasztására (feature extraction). A beszédfelismerő rendszerek fejlesztése során számos hatékony megoldás született a beszédet jellemző tulajdonságok kiválasztására, ezek közül ismertetünk most néhányat a teljesség igénye nélkül. Először bemutatjuk az általános keretet (4.1. pont). Ismertetjük a spektrális jellemzők alapján működő tulajdonság kiválasztókat, az FFT-t (4.2. pont) ehhez kapcsolódóan és spektrum pontosságára szolgáló ablak függvényeket (4.3. pont). Vizsgáljuk az MFCC-t (4.6. pont), majd a lineáris rendszereken alapuló LPC-t (4.7. pont). Felsorolás szintjén megemlítünk néhány kifinomultabb technikát, melyek a spektrális módszereken alapulnak (4.8. pont), majd végül a delta együtthatókat ismertetjük (4.9. pont).

4.1. Általános áttekintés

Az előfeldolgozó feladata a ”tiszta” digitális hullámformából a beszéd akusztikus jellemzőinek kivonása. Függetlenül attól, hogy milyen konkrét tulajdonság kiválasztó módszert használunk a beszédfelismerő rendszerek előfeldolgozói az audió jelet szeletekre, ún. ablakokra osztják. Egy megfigyelés-vektor egy ilyen ablakot fog jellemezni. A vektorok dimenziója rendszerint lényegesen kisebb, mint az ablak hossza. A jellemzők kivonása dimenziócsökkentő eljárás, ami több aspektusból is előnyös. Egyrészt egy alacsonyabb dimenziójú vektor feldolgozása kisebb műveletigénnyel jár, másrészt az így kapott megfigyelés-vektorok tárolása kisebb lemezkapacitást igényel. Elosztott feldolgozási környezetben pedig kisebb a szükséges átviteli sávszélesség, ami napjainkban, mikor egyre nagyobb teret



4.1. ábra. **Megfigyelési vektorok kivonása az audió jelből ablakozással.**

hódít az elosztott feldolgozás igen fontos szempont.

Az ablak mérete elemszámban megadva a mintavételezési frekvenciától függ. Általában, időben kifejezve az ablakméret 20ms körüli érték. Az audió jelet tehát 20ms-os szeletekre osztva vizsgáljuk. Az egyes szeleteket nem egymás után vesszük, hanem köztük átfedéssel. Az átfedés ideje lényegében megadja, milyen időközönként "gyárt" az előfeldolgozó egy megfigyelési vektort, ezt más néven megfigyelési periódusnak nevezzük. A 4.1. ábra a megfigyelési vektorok kivonásának folyamatát mutatja. Figyeljük meg az ablakméretet, az átfedést és a megfigyelési periódust.

4.2. Spektrum Analízis (FFT)

Az audió jelek döntő hányada periodikus [24] és a periodikus függvényekről J.P. de Fourier már 1822-ben megmutatta, hogy felírhatók szinusz függvények összegeként. Azaz bármely $f(t)$ periodi-

kus függvény T periódussal felírható a következő alakban:

$$f(t) = A_0 + \sum_{n=1}^{\infty} A_n sin(n\omega_p t) + B_n cos(n\omega_p t),$$

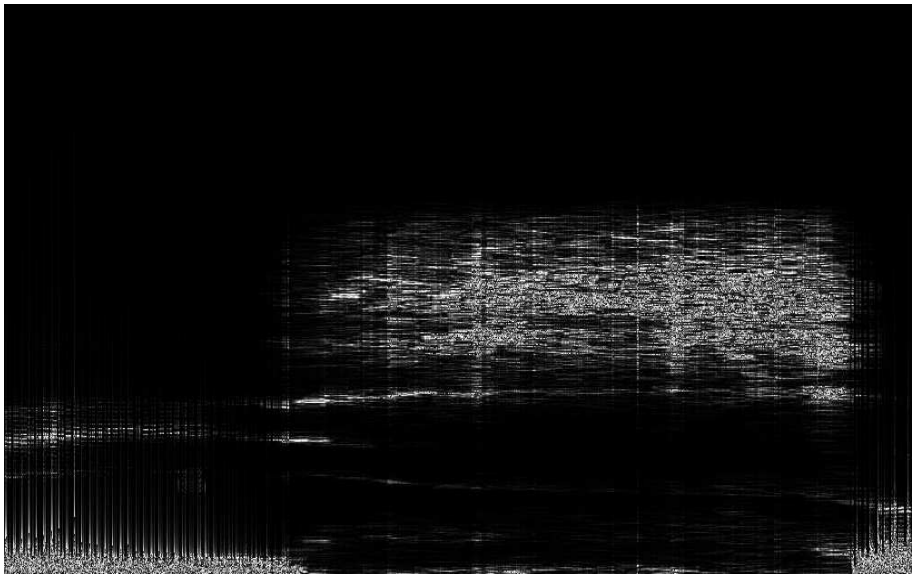
ahol $n\omega_p = \frac{n2\pi}{T}$ minden $n = 1, 2, \ldots$ -ra, és:

$$A_0 = \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} f(t)dt,$$

$$A_n = \frac{2}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} f(t)cos(n\omega_p t)dt \quad n \geq 1,$$

$$B_n = \frac{2}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} f(t)sin(n\omega_p t)dt \quad n \geq 1,$$

az A_0-t a jel átlagos értékének, az $n = 1$ -hez tartozó együtthatót alapharmónikusnak, az $n \geq 2$ -höz tartozó együtthatókat pedig felharmonikusoknak nevezünk.



4.2. ábra. **A "vissza" szó Fourier transzformáltja.**

A Fourier analízissel kapcsolatban két fontos megállapítást tehetünk:

1. Minden periodikus jel felírható egzakt matematikai formában, mert a szinusz függvény definiált.

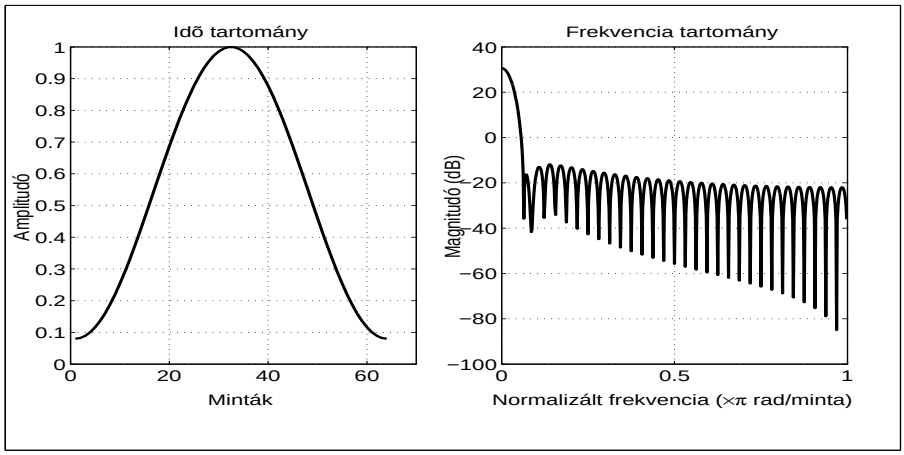
2. Az audió jelben az emberi fül is az egyes frekvencia összetevőket vizsgálja, azaz hasonló eljárást végez, mint a Fourier transzformáció. A beszédfelismerőkben, főként a beszéd jellemző tulajdonságainak kivonásában ezért fontos szerepet játszik a Fourier analízis.

A Fourier analízis következményeként minden periodikus jel reprezentálható mind az idő, mind a frekvencia tartományban. Az időtartományból frekvencia tartományba áttérés műveletét Fourier transzformációnak nevezzük. Ezen transzformáció egy gyorsított változata a Gyors Fourier Transzformáció (FFT) használatos a beszédfelismerő rendszerekben. Az algoritmus részleteit lásd [3].

A "vissza" szó frekvencia tartomány beli képét láthatjuk a 4.2. ábrán. Az analóg jel 44kHz-en lett mintavételezve, a kép 2048 elemszámú ablak méret és 512 elemszámú ablakátfedés mellett készült az FFT algoritmussal. A kép x tengelye az időt, az y a frekvenciát reprezentálja. Látható, hogy az "ssz" hang mennyivel magasabban helyezkedik el a spektrumban, mint a szó többi hangja.

4.3. Ablak függvények (window functions)

Láttuk, hogy az egyes megfigyelési vektorok mindig az aktuális ablakot jellemzik. Egy ilyen ablak feldolgozásakor a legtöbb esetben a spektrum meghatározása az első lépés.



4.3. ábra. **Hamming ablak az idő és a frekvencia tartományban.**

A Fourier transzformáció eredményében elveszik az időre vonatkozó információ, a transzformáció a feldolgozandó jelet időben

végtelen kiterjedésűnek tekinti és az, hogy a jel csak egy véges szeletét vesszük, olyan, mintha előtte egy szűrőt alkalmaznánk az időtartományban. Azonban ennek a szűrőnek a spektrumban is hatása lesz.

Az ablak függvények célja, ezen szűrő spektrum beli hatásának semlegesítése. A transzformációt még az időtartományban hajtjuk végre és a cél a spektrum kitisztítása. A transzformáció egy lineáris FIR szűrő alkalmazása, ami nem más mint egyszerűen a megfelelő indexű elemek összeszorzása. Az ablak függvények egy a beszédfelimerésben gyakran használatos osztálya a következő általános alakban írható fel:

$$w(n) = \alpha + (1 - \alpha)\cos(\frac{2\pi n}{N-1}) \quad 0 \leq n \leq N-1,$$

ahol N az ablak mérete. Két legelterjedtebb speciális esete a Hahn ablak $\alpha = 0.5$ és a Hamming ablak $\alpha = 0.54$ értékekkel. A Hamming ablakot szemlélteti a 4.3. ábra mind az idő, mind a frekvencia tartományban.

4.4. Spektrum transzformációk (warping)

A Fourier transzformáció eredményeként ismerjük az egyes frekvencia-összetevőket. Az emberi fül azonban nem lineáris skálán "figyeli" a spektrumot, ezért a spektrum transzformáló (warping) függvények a spektrumot egy nem lineáris skálára képezik. A spektrum transzformátorokat szűrőrendszerek kialakítására használják, az emberi hallás modellezésének céljából.

4.4.1. Bark Scale, Critcal Band Rate

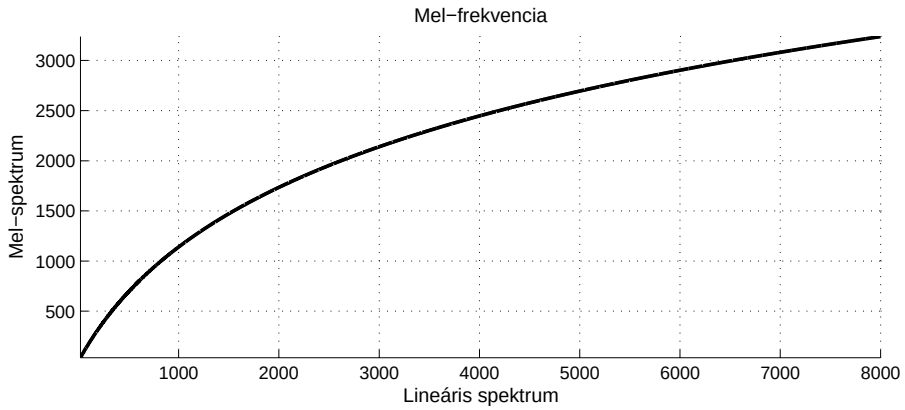
A Bark frekvencia transzformáció a következő leképezés alapján felelteti meg az f frekvenciát a b bark skálának:

$$b = 13\arctan(\frac{0.76f}{1000}) + 3.5\arctan((\frac{f}{7500})^2)$$

4.4.2. Mel Scale Filter Bank (MSFB)

Másik igen gyakran használt függvény a mel frekvencia transzfromáció. Az f frekvenciának a $\hat{f}$ mel-scale frekvencia a következő leképezés alapján fele meg:

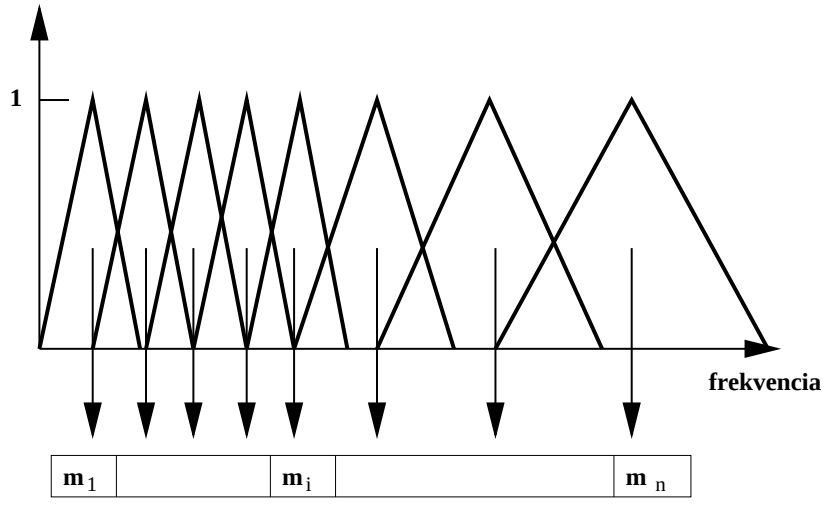
$$\hat{f} = 2959\log_{10}(1 + \frac{f}{700})$$



4.4. ábra. **Mel-spektrum.**

A lineáris skála és a mel-scale kapcsolatát mutatja 4.4. ábra.

A mel-spektrum 1kHz alatt lineáris. 1kHz felett a mel-spektrumban lineáris lépésköz exponenciálisan nővő lépést jelent a lineáris spektrumban, ez tükrözi az emberi hallás logaritmikus skáláját. A Mel Scale Filter Bank a mel-spektrumra épülő szűrőrendszer, amely a mel-spektrumban lineáris felosztással intervallumokat vesz, minden egyes intervallum középső frekvenciáját számítja legnagyobb súllyal és a szomszédos intervallumok középfrekvenciájáig az együtthatók lineárisan eltűnnek. A 4.5. ábra a szűrőrendszer alakját szemlélteti a lineáris spektrum szerint.



4.5. ábra. **Mel Scale Filter Bank a lineáris spektrumban.**

Kísérleteket folytattak arra vonatkozóan, hogy extrém külső tényezők hogyan befolyásolják az optimális spektrum transzformációt [7]. Például stressz hatása alatt lévő emberek beszédének felismerésére a mel spektrum egy módosított változatát ajánlják:

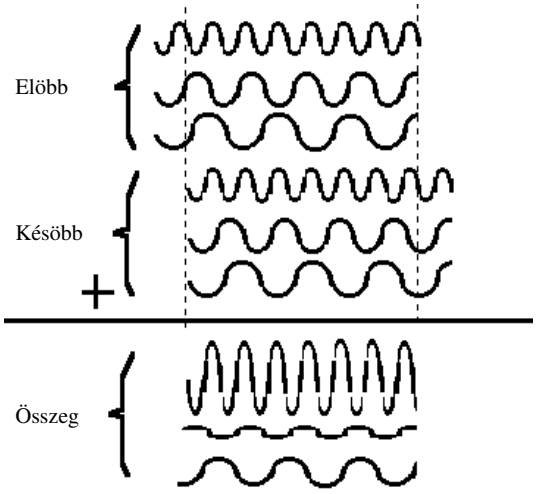
$$\hat{f} = 3070 \log_{10}(1 + \frac{f}{1000})$$

amit a stressz hatására az egyes betűk formánsaink spektrális eltolódásával magyaráznak. Ez is azt mutatja, hogy ezen módszerek még messze nem kifinomultak.

Az MSFB szűrőrendszerben a szomszédos szűrők 50%-os átfedéssel nyúlnak egymásba. Az emberi hallást leginkább modellező optimális átfedését vizsgálták a [38]. közleményben. A javasolt átfedés az 50%-os helyett 80%.

4.5. Kepsztrum

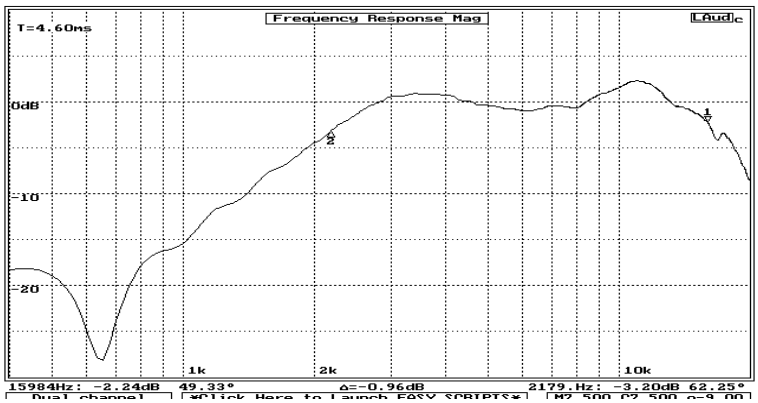
A "kepsztrum" szó a spektrum anagrammája, először Bogert használta 1963-ban a [8] közleményében. A kepsztrum a spektrum spektrumaként értelmezhető. A kepsztrum x tengelye (a kefrencia) időegységekből áll, de a hozzájuk tartozó értékek a spektrum változását jelentik.



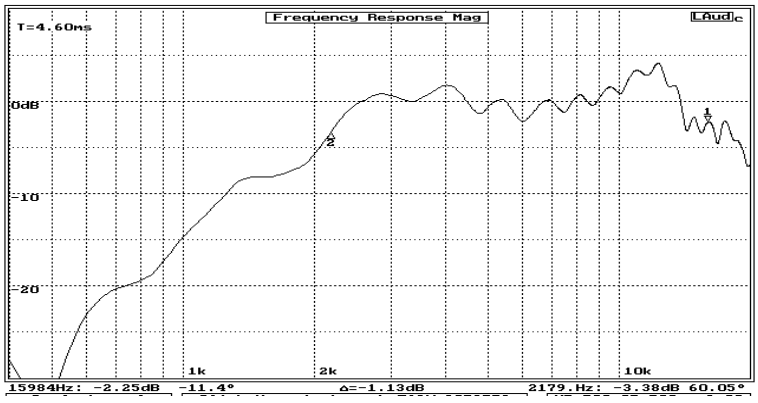
4.6. ábra. **Különböző frekvenciájú szinuszok szuperpozíciója.**

A következőkben bemutatjuk a kepsztrum egy érdekes tulajdonságát. Tegyük fel most, hogy két szinuszhullám, amely azonos frekvenciájú, két különböző időpillanatban érkezik egy mikrofonra. Amennyiben a fázisuk megegyezik, akkor felerősítik, ha a fázisuk

ellenkező, akkor kioltják egymást. A 4.6. ábrán különböző frekvenciájú szinusz hullámokat láthatunk, amelyek azonos időeltolodással érkeznek egy vételi pontra. Az ábra alsó harmadán az azonos frekvenciájú párok szuperpozíciója látható. Vegyük észre, hogy frekvencia változásával a szuperpozíció jellege is változik, az első esetben a hullámok felerősítik, míg a középsőben kioltják egymást. Ez a jelenség a frekvencia változásával periodikusan teljesül.



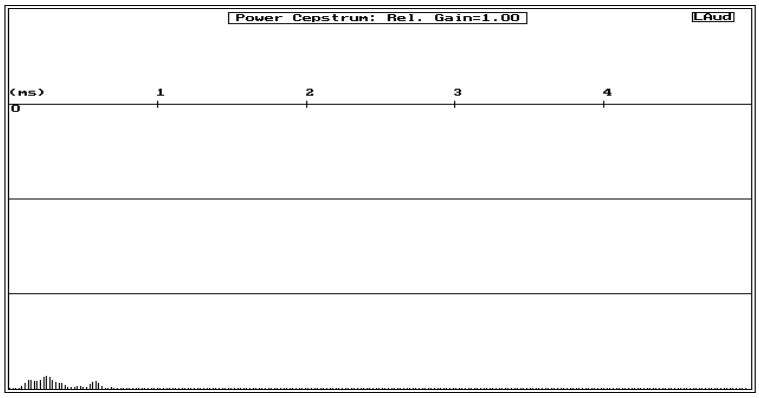
(a) Kürt hangjának spektruma tiszta környezetben.



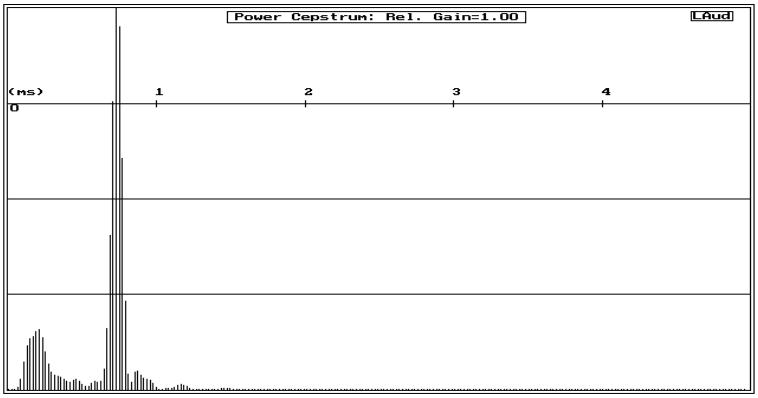
(b) Kürt hangjának spektruma visszhangos környezetben.

4.7. ábra. **Kürt hangjának spektruma visszhangos és tiszta környezetben.**

A 4.7. ábra egy kürt hangjának spektrumát mutatja tiszta és visszhangos környezetben. Figyeljük meg a visszhangos esetben a spektrális kép jobb oldalán látható ”fodrozódást”.



(a) Kürt hangjának kepsztruma tiszta környezetben.



(b) Kürt hangjának kepsztruma visszhangos környezetben.

4.8. ábra. **Kürt hangjának kepsztruma visszhangos és tiszta környezetben.**

A 4.8. ábrán a 4.7. ábra kürt hangjának kepsztruma látható mind a tiszta, mind a visszhangos esetben. Figyeljük meg, hogy a visszhangos környezet hogy befolyásolta a kepsztrum alakulását. A Fourier analízis értelmében bármely hangot tekinthetjük végtelen sok szinusz hullám szuperpozíciójának és ahogy a 4.6. ábrán láttuk, a visszhang a spektrumban periodicitást jelent, ezzel áll összhangban a 4.8(b). ábra is.

A kepsztrum beszédmodellben azzal a feltételezéssel élnek, hogy a tiszta beszéd hangot az átviteli csatorna (a száj és a környezet) mint egy lineáris szűrő teszi zajossá [26]. Az ilyen lineáris szűrők az időtartományban a konvolúció műveletével írhatók le. Az időtartományban a jelet ezen két elválasztható összetevő konvolúciója szerint értelmezve:

$$X(t) = \int_0^t G(t)H(t-\tau)d\tau$$

az időtartomány beli konvolúció szorzás a frekvencia tartományban:

$$X(\omega) = G(\omega)H(\omega)$$

majd az abszolút érték logaritmusát véve, aminek hatására a szorzat összegre bomlik:

$$log|X(\omega)| = log|G(\omega)| + log|H(\omega)|$$

a kapott jel alakját az alacsony frekvenciás, míg a finom részleteket a magas frekvenciás összetevők tartalmazzák. Ezek szétválasztására alkalmazzák az inverz Fourier transzformációt:

$$F^{-1}(log|X(\omega)|) = F^{-1}(log|G(\omega)|) + F^{-1}(log|H(\omega)|),$$

ahol F a Fourier transzformációt jelöli. A kepsztrum számításának lépései sorrendben a következők:

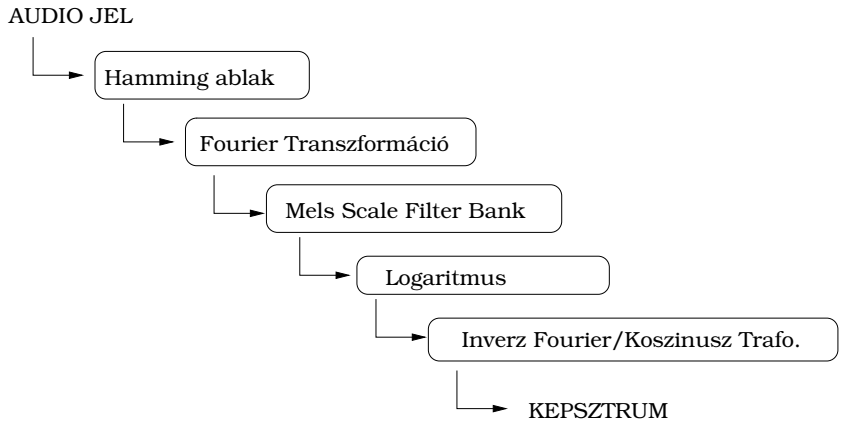
1. Hamming ablak
2. FFT
3. Logaritmus
4. Inverz FFT

4.6. Mel Scale Cepstral Coefficients (MFCC)

A Mel Scale Kepsztrum a Mel Scale Filter Bank és a kepsztrum együttes alkalmazása, ebben az esetben a számítás menetét a 4.9. ábra szemlélteti. Az így nyert együtthatókat a mel-frekvenciás kepsztrális koefficienseknek nevezik. Általában 13 ilyen együtthatót határoznak meg, amihez első és másodrendű delta együtthatókat számítanak, így kapjuk a manapság leggyakrabban használt front-end-ek által előállított 39 dimenziós megfigyelésvektorokat.

4.7. Lineáris Predikciós Együtthatók (LPC)

Az LPC technológiát eredetileg az audiójel minél kisebb sávszélességen való átviteléhez, azaz minél kevesebb biten történő ábrázolásához fejlesztették ki, de a beszédfelismerésben is alkalmazzák. Az alapötlet a módszer mögött a következő, egy adott pillanatban az audió jel értékét megpróbálja az azt megelőző értékek lineáris kombinációjaként közelíteni. A jel egy rövidebb szakaszán, az adott ablakban, az aktuális érték és a jósolt érték közti hiba minimalizálásával meghatározhatók az ablakot jellemző predikciós együtthatók.



4.9. ábra. **A mel-scale kepsztrum számításának lépési.**

Az LPC az n. időpillanatban az $s(n)$ mintát a következő képpen közelíti:

$$s(n) \approx a_1 s(n-1) + a_2 s(n-2) + \ldots + a_p s(n-p),$$

ahol p a közelítés rendje. A jósolt érték:

$$\hat{s}(n) = \sum_{k=1}^{p} a_k s(n-k)$$

Az adott pillanatban a jóslási hiba értéke:

$$e(n) = s(n) - \hat{s}(n) = s(n) - \sum_{k=1}^{p} a_k s(n-k)$$

A cél a jel egy adott szakaszán (ablakán) olyan a_k együtthatók keresése, amire a négyzetes hiba minimális:

$$E_n = \sum_m e_n^2(m) = \sum_m (s_n(m) - \sum_{k=1}^{p} a_k s_n(m-k))^2,$$

ahol $s_n(m) = s(n+m)$. Az E_n-t minimalizáló együtthatókra teljesül, hogy $\frac{\delta E_n}{\delta a_i} = 0$ minden $i = 1, 2, \ldots, p$-re. A derivált számítása után a következő rendszerhez jutunk:

$$\sum_m s_n(m-i)s_n(m) = \sum_{k=1}^{p} a_k' \sum_m s_n(m-i)s_n(m-k) \quad 1 \leq i \leq p,$$

ahol a_k' az optimális együtthatókat jelöli. Adott p egyenlet p ismeretlennel. Az együtthatók meghatározására javasolt eljárások az autókorrelációs és a kovariancia módszer [18].

4.8. Egyéb spektrális módszerek

A spektrum alapú jellemző kivonás a mai napig a kutatások pezsgő szintere. A cél minél robusztusabb, a zajra és a környezet hatásaira kevésbé érzékeny módszerek fejlesztése. A legtöbb módszer ezek közül a spektrum feldolgozásában különbözik, rendszerint a frekvencia transzformáló leképezés és a szűrőrendszer finomítása a járt út. A Hermansky által közölt Perceptual Linear Prediction (PLP) módszer [16] és annak további finomítása a RASTA-PLP, vagy a Skowronksi által közölt Human Frequency Cepstral Coefficients (HFCC) eljárás [37] is mind ezt a célt szolgálják.

4.9. Első és másodrendű delta együtthatók

Egy tipikus befejezés az előfeldolgozó részéről az elsőrendű delta és a másodrendű (delta-delta) együtthatók számítása. Ez nagyban javítja a HMM akusztikus modellen alapuló felismerőrendszereket, mert így az adott időpillanatot jellemző megfigyelési értékek információt hordoz a környezetéről is. A delta együttható számítási szabálya:

$$\Delta o_t = \frac{\sum_{\tau=1}^{T_r} \tau (o_{t+\tau} - o_{t-\tau})}{2 \sum_{\tau=1}^{T_r} \tau^2},$$

ahol $2T_r + 1$ a regressziós ablak mérete, a delta-delta $\Delta^2 o_t$ együtthatókat ugyanezzel az eljárással számítják a delta együtthatókra. Tipikusan a $T_r = 2$, így a másodrendű delta együtthatók 9 megfigyelésnyi távolságot ölelnek át. Általában 13 dimenziós megfigyelési vektorokat használnak, amihez hozzáadva a delta és a delta-delta együtthatókat a dimenzió 39 lesz.

4.10. Vektor kvantálás (VQ)

A vektor kvantálás (Vector Quantization, azaz VQ) a 2.2.2. pontban látott eljárás megfelelője magasabb dimenzióban. Az egy dimenziós esetet az irodalomban skalár-, míg magasabb dimenzióban vektor-kvantálásnak nevezik. Az előfeldolgozók általános leírásában említettük (4.1. pont), hogy a dimenziócsökkentés több szempontból is fontos. A vektor-kvantálás ezen gondolatmenet továbbgöngyölítése. Tekintsük a következő példát, tegyük fel, hogy a digitális audió 10kHz-en mintavételezett és 16 bites kvantálással. A "tiszta" hullámforma így 160kb-et igényel másodpercenként, tegyük fel továbbá, hogy az előfeldolgozó tíz dimenziós vektorokat gyárt 10ms-os megfigyelési periódussal és minden komponenst 16biten ábrázol. Ez azt jelenti, hogy másodpercenként 16kb-en jellemezzük a jelet, ami már csak 1/10-e az eredeti reprezentációnak. A célunk beszéd ábrázolása, ezért vesszük a tanítóhalmaz beli szavak fonémáihoz tartozó megfigyelés vektorokat. Ezt a továbbiakban kódkönyvnek nevezzük (codebook). Például álljon a kódkönyv 1024 vektorból (az angol nyelvben ez kb. 25 variánst jelent a 40 különböző fonémához [1]) Ezek után egy megfigyelési vektor feljegyzéséhez elég a kódkönyv hozzá legközelebb álló elem indexének feljegyzése ami ebben az esetben egy 10 bites érték, ami további 1/16-os csökkentése a szükséges kapacitásnak. A következő néhány pontban összefoglaljuk a VQ eljárás előnyeit és hátrányait.

A módszer előnyei, hogy

- lényegesen csökkenti a megfigyelési vektorok ábrázoláshoz szükséges kapacitás igényt.
- csökkenti a megfigyelési vektorok közötti hasonlóság meghatározásához szükséges számítást. A hasonlóság mértékét a kódkönyv beli vektorok közti hasonlóságok adják, amelyek a kódkönyv meghatározása után előre számíthatók.

A módszer hátrányai, hogy

- miután a kódkönyv véges sok értéket tartalmaz, így az aktuális megfigyelési vektor legmegfelelőbb reprezentálása kvantálási hibával, azaz a reprezentáció torzulásával jár.
- a kódkönyv tárolása és hatékony kezelése nem triviális, minél több elemet tartalmaz a könyv annál kisebb a kvantálási hiba, viszont nő a legmegfelelőbb elem megkeresésének műveletigénye.

Matematikailag a vektor kvantálás a következőképpen írható fel, adott d dimenziós x folytonos megfigyelési vektorhoz egy d dimenziós

z diszkrét értékű vektort kell rendelni. Azaz,

$$z = q(x),$$

ahol

- $q(.)$ a kvantálási operátor.
- $x = (x_1, \ldots, x_d)^T \in \mathbb{R}^d$ a meggfigyelési vektor.
- $z = (z_1, \ldots, z_d)^T \in Z = \{z_i, 1 \leq i \leq L\}$ kódkönyvnek, ahol $z_i = (z_{i_1}, \ldots, z_{i_d})^T$

Az L értékét a kódkönyv méretének, vagy szintjeinek nevezik. A kódkönyv elkészítéséhez a d dimenziós tér egy diszkretizálása szükséges.

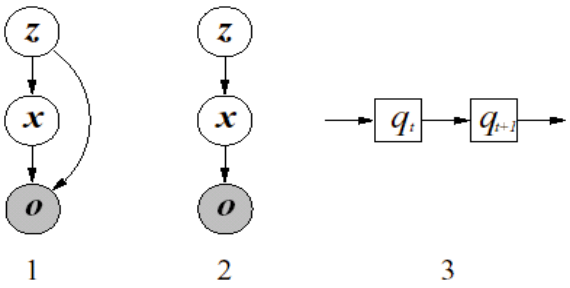
5. fejezet

Akusztikus modellek és algoritmusaik

Ebben a fejezetben az akusztikus modellekkel foglalkozunk. Részletesen ismertetjük a Rejtett Markov Modelleket, mint a legszélesebb körben alkalmazott akusztikus modellt, majd említés szintjén vizsgálunk más, alternatív lehetőségeket.

5.1. Áttekintés, Bayes-hálók

Korábban láttuk, hogy a felismerő feladata a 3.2. egyenlet hatékony megoldása, melyben az akusztikus modell a $P(O|W)$ valószínűség számításáért felelős. Azaz annak a meghatározásáért, hogy a feltételezett szósorozat "kimondása" milyen valószínűséggel eredményezi a vizsgált megfigyelés sorozatot. Az akusztikus modell a megfigyelés sorozat generatív oldalról való megközelítése. Az akusztikus modell célja a beszéd belső állapotainak megfogalmazása, majd az ezen



5.1. ábra. **Valószínűségi függőségek szemléltetése Bayes-hálókkal.**

állapotok valamilyen sorozatán haladva az adott sorozathoz kapcsolódó megfigyelési értékek generálása. A hatékony felismerő algoritmus természetesen nem az összes lehetséges állapot sorozathoz generálható megfigyelés sorozatot hasonlítja az előfeldolgozótól kapott megfigyeléssorozattal, hanem a kapott megfigyeléssorozat alapján a legjobban illeszkedő belső állapotsorozatot keresi.

Az egyes állapotsorozatokban az egymást követő állapotok és a hozzájuk tartozó megfigyelések statisztikailag függnek egymástól. Valószínűségi változók egymástól való függőségeinek reprezentálására alkalmas eszköz a Bayes-háló. Tekintsük a 5.1. ábrát. (Az ábrán a szögletessel a diszkrét, míg körrel a folytonos értékű valószínűségi változót jelöljük, továbbá akusztikus modellek ábrázolása esetén sötéttel jelöljük a "megfigyelhető" értékeket és világossal a belső változókat.) Az első és a második példában három valószínűségi változót, z-t, x-et és o-t láthatunk. Az első esetben o mind x-től, mind z-től statisztikailag függ, együttes eloszlásuk a következő alakban írható:

$$p(o, x, z) = p(x)p(z|x)p(o|z, x)$$

míg a második esetben o már csak x-től, azaz:

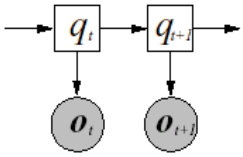
$$p(o, x, z) = p(x)p(z|x)p(o|x)$$

Az ábra harmadik példája azt az összefüggést szemlélteti, hogy a $Q = \{q_1, \ldots, q_T\}$ rendszerre, a $t + 1$.-ik időpillanatban teljesül, hogy:

$$P(q_{i_1}, q_{i_2}, \ldots, q_{i_{t+1}}) = P(q_{i_{t+1}}|q_{i_t})$$

vagyis a $t + 1$.-ik időpillanat beli érték csak a t.-ik pillanat beli értéktől függ, ez a Markov feltevés. A Bayes-háló alkalmas eszköz az akusztikus modell szerkezetének szemléltetésére.

Az első akusztikus modell, amit részletesen tárgyalunk az Rejtett Markov Modell. Egy ilyen rendszer Bayes hálóját szemlélteti a 5.2. ábra.



5.2. ábra. **Rejtett Markov Modell szemléltetése Bayes-hálóval.**

5.2. Rejtett Markov Modellek

A Rejtett Markov Modell a beszédfelismerésben a legszélesebb körben alkalmazott akusztikus modell, ezért ebben a fejezetben részletesen ismertetjük. Először definiáljuk a Markov folyamatokat (5.2.1 Alfejezet). Ezután a rejtett Markov modelleket értelmezzük (5.2.2 Alfejezet), majd tekintjük a hozzá kapcsolódó három alapproblémát (5.2.3 Alfejezet) és annak megoldásait. Formálisan ismertetjük a Viterbi algoritmust (5.2.4 Alfejezet), majd a Baum-Welch módszert, mint a HMM egy tanítási lehetőségét (5.2.4 Alfejezet). Végül a modellt kiterjesztjük a folytonos megfigyelési értékek esetére (5.2.5 Alfejezet). A fejezet Rabiner [31] közleményének Szendrő Balázs fordítása [4] alapján készült.

5.2.1. Diszkrét Markov folyamatok

Tekintsünk egy folyamatot, amely minden $t = 1, 2, \dots$ időpillanatban véges sok állapot valamelyikében tartózkodik. Az állapotok számát jelölje N, az állapotokat pedig $S_1, S_2, \dots, S_N$. Tetszőleges t időpillanatban az aktuális állapotot jelölje q_t. A rendszer minden időpillanatban egy állapotátmeneten megy keresztül, melyet az állapotokhoz tartozó átmeneti valószínűségek határoznak meg.

Általánosan egy ilyen rendszer egy adott t pillanatbeli állapotát az határozza meg, hogy a t-t megelőző időpillanatokban milyen állapotokban volt. Speciálisan, ha ez az állapot csak az őt közvetlenül megelőző állapottól függ, akkor azt mondjuk, hogy a folyamat elsőrendű Markov folyamat, azaz teljesül hogy:

$$P(q_t = S_j | q_{t-1} = S_i, q_{t-2} = S_k, \dots) = P(q_t = S_j | q_{t-1} = S_i). \qquad (5.1)$$

A továbbiakban olyan folyamatokkal foglalkozunk amelyekre a (5.1) egyenlet minden t időpillanatban igaz.

Az állapotok közötti átmenet-valószínűségekre vezessük be a következő jelölést:

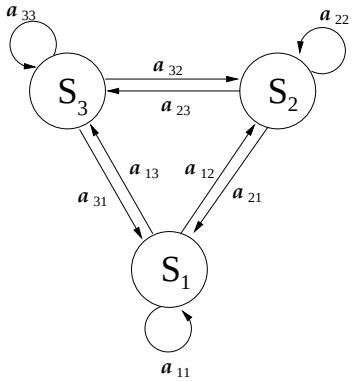
$$a_{ij} = P(q_t = S_j | q_{t-1} = S_i), \quad 1 \leq i, j \leq N$$

, melyekre teljesül:

$$a_{ij} \geq 0$$

$$\sum_{j=1}^{N} a_{ij} = 1$$

Ami tehát azt jelenti, hogy az átmenet-valószínűségek az egyes állapotokban érvényes diszkrét eloszlások. Példaként tekintsük a 5.3. ábrát.



5.3. ábra. **3 állapotú Markov modell.**

5.2.2. Rejtett Markov Modellek

Eddig olyan Markov modellt tekintettünk, ahol az állapotok teljesen megfigyelhetőek, illetve egyértelműen azonosíthatóak. A gyakorlatban ez ritkán van így. Csupán azt feltételezhetjük, hogy a megfigyelt értékek mögött egy Markov tulajdonságú folyamat húzódik meg. Ebben az esetben a megfigyelési értékek a belső állapotok valamilyen valószínűségi függvényeként tekinthetők, vagyis a háttérben zajló Markov folyamat minden egyes időpillanatban egy megfigyelési értéket bocsájt ki. A gyakorlatban csak a megfigyelési értékek láthatóak, a mögöttük zajló folyamat rejtve marad, innen a név, Rejtett Markov Model (HMM) [31], [30].

Formálisan a következő mennyiségekkel írható le egy HMM:

1. N, a modell rejtett állapotainak száma. A gyakorlatban ezt a számot gyakran valamilyen - a modellezni kívánt folyamatra vonatkozó - ismeretünk alapján adjuk meg.

2. M, a különböző megfigyelhető szimbólumok száma. A modellezni kívánt folyamat kimenetének megfelelően választott diszkrét ábécé elemeinek száma. Az ábécére vezessük be a $V = \{v_1, v_2, \ldots, v_M\}$ jelölést.

3. $A = \{a_{ij}\}$, az állapot-átmeneti eloszlásmátrix, ahol

$$a_{ij} = P(q_{t+1} = S_j | q_t = S_i), \quad 1 \leq i, j \leq N.$$

   Speciálisan, ha $\forall i, j : a_{ij} > 0$, akkor minden állapotból minden állapotot elérhetünk a HMM-ben[1]. Léteznek olyan modellek is, amelyekben csak bizonyos állapotok érhetőek el más állapotokból. (Pl. ilyenek a "balról-jobbra" modellek.)

4. $B = \{b_j(k)\}$, a megfigyelt (kimeneti) szimbólumok eloszlása a j állapotban, azaz

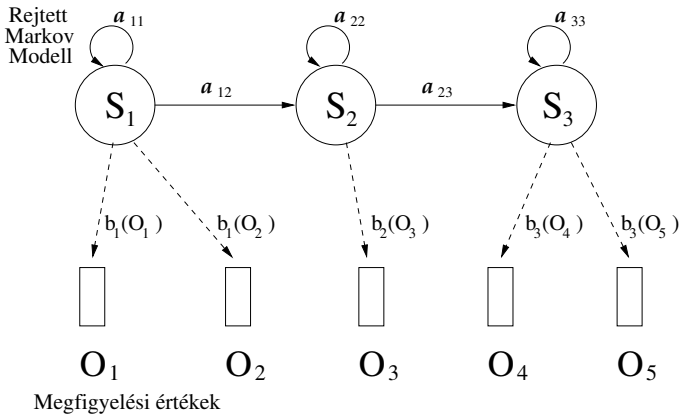
$$b_j(k) = P(v_k \text{ a megfigyelés} | q_t = S_j), \quad \begin{array}{l} 1 \leq j \leq N \\ 1 \leq k \leq M. \end{array}$$

5. $\pi = \{\pi_i\}$ a kezdeti-állapot valószínűségek, azaz

$$\pi_i = P(q_1 = S_i), \quad 1 \leq i \leq N.$$

Tekintsük a következő megfigyeléssorozatot:

$$O = O_1 O_2 \ldots O_T$$



5.4. ábra. **Rejtett Markov Modell.**
Három rejtett állapottal rendelkező balról-jobbra modell megfigyeléseket generál.

Ahol T a megfigyelések száma, és $\forall t \in \{1, \ldots, T\} : O_t \in V$. A modell N, M, A, B és π paramétereinek ismeretében a következő algoritmussal generálhatunk ilyen megfigyeléssorozatot:

1. Válasszunk egy kezdeti $q_1 = S_i$ állapotot a π kezdeti-állapot valószínűségek alapján, és legyen $t = 1$.
2. Válasszunk egy $O_t = v_k$ kimeneti értéket az S_i állapotban érvényes $b_i(k)$ valószínűséggel.
3. Lépjünk a következő $q_{t+1} = S_j$ állapotba az állapot-átmeneti eloszlásmátrix S_i állapotban érvényes a_{ij} valószínűségével.
4. Legyen $t = t + 1$, és a 2. lépéstől folytassuk az eljárást, amíg $t < T$.

A fenti tárgyalásból látható, hogy egy HMM teljesen megadható az N, M paraméterekkel, a kimeneti értékek V ábécéjével, és az A, B, illetve a π valószínűségi eloszlásokkal. A tömörség érdekében vezessük be a

$$\lambda = (A, B, \pi)$$

jelölést a modell paramétereinek leírására.

[1] Az ilyen modelleket ergodikus modelleknek nevezik.

5.2.3. A HMM három alapproblémája

Ahhoz, hogy HMM-eket illeszthessünk megfigyelt folyamatokhoz, három alapkérdést kell megválaszolni:

1. Adott $O = O_1O_2 \dots O_T$ megfigyeléssorozat, és $\lambda = (A, B, \pi)$ modell esetén hogyan tudjuk hatékonyan kiszámítani a $P(O|\lambda)$ valószínűséget?

2. Az $O = O_1O_2 \dots O_T$ megfigyeléssorozat, és $\lambda = (A, B, \pi)$ modell ismeretében melyik az a $Q = q_1q_2 \dots q_T$ belső állapotsorozat, amelyik valamilyen értelemben a legjobban leírja a megfigyelt sorozatot?

3. Hogyan maximalizáljuk a $P(O|\lambda)$ valószínűséget a modell $\lambda = (A, B, \pi)$ paramétereinek függvényében?

Az 1. kérdést kiértékelési problémának is nevezik. Hogyan számíthatjuk ki a $P(O|\lambda)$ valószínűséget, ha ismerjük O-t és λ-t? A kérdést úgy is feltehetjük, hogy a modell mennyire jól írja le a megfigyeléssorozatot, vagyis mennyire jól illeszkedik. Ez utóbbi különösen fontos, hiszen e probléma megoldása lehetővé teszi, hogy több modell esetén kiválasszuk a megfigyeléseinkhez legjobban illeszkedőt.

A 2. kérdés a modell belső állapotainak felfedésére vonatkozik, vagyis arra, hogy megtaláljuk a "pontos" állapotsorozatot, amely a megfigyelést generálta. Látnunk kell azonban, hogy ilyen "pontos" állapotsorozatot - a degenerált modellek kivételével - nem tudunk megadni. Ezért a gyakorlatban a konkrét alkalmazástól függő optimalitási kritériumokat vezetnek be az állapotsorozatokra. Ennek a kérdésnek a megválaszolásával következtethetünk például a modell belső szerkezetére, egy adott megfigyeléssorozathoz tartozó optimális állapotsorozatra, stb.

A 3. kérdés a modellillesztési feladat, azaz egy rögzített O megfigyeléssorozat birtokában keressük a modell $\lambda = (A, B, \pi)$ paramétereinek azon értékét, amelyre a $P(O|\lambda)$ valószínűség maximális. Ez tulajdonképpen a HMM rátanítását jelenti az O példahalmazra. A kérdés az, hogy milyen algoritmust tudunk adni erre a tanítási feladatra.

5.2.4. A HMM három alapproblémájának megoldásai

A következőkben formális megoldást adunk a három alapproblémára. Az algoritmusok programozásához még számos numerikus problémát kell megoldani, amelyekkel itt nem foglalkozunk.

Az 1. probléma megoldása

Ismerjük tehát az $O = O_1 O_2 \dots O_T$ megfigyeléssorozatot, és a $\lambda = (A, B, \pi)$ modellparamétereket. Feladatunk a $P(O|\lambda)$ valószínűség kiszámítása. A legkézenfekvőbb módszer, ha az összes lehetséges Q állapotsorozatra összeadjuk a $P(O|Q)P(Q)$ értékeket. Tekintsünk egy

$$Q = q_1 q_2 \dots q_T \tag{5.2}$$

állapotsorozatot, ahol q_1 a kezdeti állapot. Az O megfigyeléssorozat (5.2) állapotsorozathoz tartozó valószínűsége

$$P(O|Q, \lambda) = \prod_{t=1}^{T} P(O_t|q_t, \lambda)$$

alakú, ahol feltettük, hogy a megfigyeléseink függetlenek. Így kapjuk, hogy

$$P(O|Q, \lambda) = b_{q_1}(O_1) \cdot b_{q_2}(O_2) \cdots b_{q_T}(O_T).$$

A (5.2) állapotsorozat valószínűsége

$$P(Q|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \cdots a_{q_{T-1} q_T}$$

, O és Q együttes valószínűsége pedig a fenti kettő szorzata lesz. Ez tehát annak a valószínűsége, hogy a Q állapotsorozaton haladt a rendszer és az O kimenetet adta:

$$P(O, Q|\lambda) = P(O|Q, \lambda)P(Q, \lambda)$$

Az O kimenet valószínűségét kiszámíthatjuk az O és Q együttes valószínűségének Q szerinti összegzésével, így

$$\begin{aligned} P(O|\lambda) = \sum_{Q} & P(O|Q, \lambda)P(Q|\lambda) \\ & \sum_{q_1, q_2, \dots, q_T} \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) \\ & \cdots a_{q_{T-1} q_T} b_{q_T}(O_T). \end{aligned} \tag{5.3}$$

A $P(O, Q|\lambda)$ kiszámítását a fentiek alapján a következőképpen interpretálhatjuk. Kezdetben ($t = 1$) a q_1 állapotban vagyunk π_{q_1} valószínűséggel, és az O_1 szimbólumot generáljuk $b_{q_1}(O_1)$ valószínűséggel. A következő pillanatban ($t = t+1$) átlépünk q_1-ből a q_2 állapotba $a_{q_1 q_2}$ valószínűséggel, majd generáljuk O_2-t $b_{q_2}(O_2)$ valószínűséggel, majd így tovább, egészen az utolsó szimbólum generálásáig.

Bár a leírt algoritmus matematikailag korrekt, a $P(O|\lambda)$ kiszámítása a definíció alapján kivitelezhetetlen, hiszen nagyságrendileg $2T \cdot N^T$ műveletet igényel: A lehetséges T hosszú állapotsorozatok száma N^T, és minden ilyen sorozatra kb. $2T$ műveletet kell elvégezni az (5.3) összegzésben. (Egészen pontosan $(2T-1)N^T$ szorzásra, és $N^T - 1$ összeadásra van szükség.)

A következőkben ismertetünk egy hatékony eljárást $P(O|\lambda)$ kiszámítására, amelyet "előre-hátra" (Forward-Backward) módszernek hívnak [5], [6]. Vezessük be az

$$\alpha_t(j) = P(O_1 O_2 \cdots O_t, q_t = S_j|\lambda)$$

változót, azaz $\alpha_t(j)$ jelentse annak valószínűségét, hogy eddig az $O_1 O_2 \ldots O_t$ sorozatot figyeltük meg, és jelenleg (a t pillanatban) az S_j állapotban vagyunk. Könnyen belátható, hogy az $\alpha_t(j)$ kiszámítható az

$$\begin{aligned} \alpha_1(j) &= \pi_j b_j(O_1), & 1 \le j \le N \\ \alpha_{t+1}(j) &= \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}), & 1 \le t \le T-1 \\ & & 1 \le j \le N \end{aligned}$$

egyenletekkel megadott rekurzióval, és

$$P(O|\lambda) = \sum_{j=1}^{N} \alpha_T(j).$$

Ez utóbbi kiszámítása nagyságrendileg N^2T műveletet igényel, ami jelentős javulás a $2T \cdot N^T$-hez képest.

A 2. és 3. problémák megoldásához szükségünk lesz még egy olyan változó bevezetésére, amely a hátralévő $O_{t+1} O_{t+2} \ldots O_T$ megfigyeléssorozat $q_t = S_i$ feltételre vonatkozó valószínűségét adja meg. Formálisan

$$\beta_t(i) = P(O_{t+1} O_{t+2} \cdots O_T | q_t = S_i, \lambda).$$

A $\beta_t(i)$ értékeinek kiszámításához szintén felírhatunk egy rekurziót:

$$\begin{aligned} \beta_T(i) &= 1, & 1 \le i \le N \\ \beta_t(i) &= \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j), & t = T-1, T-2, \ldots, 1 \\ & & 1 \le i \le N. \end{aligned}$$

A 2. probléma megoldása, Viterbi algoritmus

Az 1. problémára adható egzakt megoldással szemben, a 2. problémára, vagyis az "optimális" állapotsorozat megtalálására számos algoritmus adható attól függően, hogy mit nevezünk optimális állapotsorozatnak.

Egy lehetséges megközelítés, ha az O sorozat minden O_t tagjához egyenként választjuk ki azt az S_j állapotot, amelyre a $P(O_t|q_t = S_j)$ maximális. Az ilyen S_j állapotokat a továbbiakban *helyes* állapotoknak nevezzük. Vagyis az optimális állapotsorozat az, amelyben a helyes állapotok számának várható értéke maximális.

Ennek a megoldásához vezessük be a

$$\gamma_t(i) = P(q_t = S_i|O, \lambda)$$

jelölést, annak valószínűségét, hogy a t pillanatban S_i állapotban vagyunk, feltéve, hogy ismerjük az O sorozatot. $\alpha_t(i)$ és $\beta_t(i)$ definíciójából adódik, hogy

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(O|\lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$

és $\gamma_t(i)$ eloszlás, mert

$$\sum_{i=1}^{N} \gamma_t(i) = 1.$$

A feladat tehát azon q_t állapotok meghatározása, amelyekre

$$q_t = \arg\max_{1 \le i \le N} [\gamma_t(i)], \quad 1 \le t \le T. \tag{5.4}$$

Habár (5.4) maximalizálja a helyes állapotok számát (minden O_t tagra a legvalószínűbb S_j kiválasztásával), problémák lehetnek az így kapott állapotsorozattal. Mivel nem vesszük figyelembe az a_{ij} állapot-átmenet valószínűségeket előfordulhat, hogy a kapott optimális állapotsorozat érvénytelen, mert tartalmaz 0 valószínűségű átmenetet. (Előfordul benne S_iS_j, miközben $a_{ij} = 0$.)

Egy lehetséges megoldás, ha helyes állapotok helyett helyes állapotpárok (q_tq_{t+1}), állapot-hármasok ($q_tq_{t+1}q_{t+2}$), stb. várható értékét maximalizáljuk. Leggyakrabban a teljes Q sorozatra maximalizálják $P(Q|O, \lambda)$ értékét. Mivel O rögzített, ez megegyezik $P(Q, O|\lambda)$ maximalizálásának feladatával.

Erre egy formális megoldás a Viterbi algoritmus [43],[10]: Legyen $O = O_1O_2\cdots O_T$ rögzített megfigyeléssorozat. A hozzátartozó legvalószínűbb $Q = q_1q_2\cdots q_T$ állapotsorozat megtalálásához definiáljuk a következő mennyiséget:

$$\delta_t(i) = \max_{q_1q_2\cdots q_{t-1}} P[q_1q_2\cdots q_t = S_i, O_1O_2\cdots O_t|\lambda]$$

, azaz $\delta_t(i)$ jelöli az első t megfigyeléshez tartozó, S_i-ben végződő legvalószínűbb állapotsorozat valószínűségét. Nyilván igaz a következő:

$$\delta_{t+1}(j) = [\max_i \delta_t(i)a_{ij}] \cdot b_j(O_{t+1}). \tag{5.5}$$

Ahhoz, hogy ebből kinyerjük a helyes állapotsorozatot, fel kell jegyeznünk (5.5)-ben a $\max$ argumentumát minden t-re és j-re. Ehhez vezessük be a $\psi_t(j)$ változót. A Viterbi algoritmus tehát:

1. Inicializálás:

$$\begin{aligned}\delta_1(i) &= \pi_i b_i(O_1), \quad 1 \le i \le N\\ \psi_1(i) &= 0.\end{aligned}$$

2. Rekurzió:

$$\delta_t(j) = \max_{1\le i\le N}[\delta_{t-1}(i)a_{ij}]b_j(O_t), \qquad \begin{aligned}&2 \le t \le T\\ &1 \le j \le N\end{aligned}$$

$$\psi_t(j) = \arg\max_{1\le i\le N}[\delta_{t-1}(i)a_{ij}], \qquad \begin{aligned}&2 \le t \le T\\ &1 \le j \le N.\end{aligned}$$

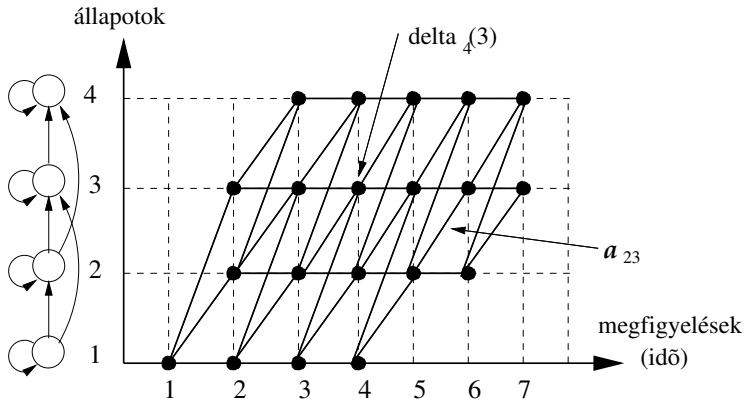
3. Terminálás:

$$\begin{aligned}p^* &= \max_{1\le i\le N}[\delta_T(i)]\\ q_T^* &= \arg\max_{1\le i\le N}[\delta_T(i)].\end{aligned}$$

4. Az állapotsorozat kinyerése:

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \quad t = T-1, T-2, \ldots, 1.$$

Az algoritmus végrehajtása után $p^* = \max_Q P(Q, O|\lambda)$, és $Q = (q_t^*)_{1\le t\le T}$ a keresett állapotsorozat, amelyre $P(Q, O|\lambda) = p^*$.



5.5. ábra. **A Viterbi algoritmus szemléltetése.**

A 3. probléma megoldása, Baum-Welch algoritmus

A HMM-ek harmadik, és egyben legnehezebb problémája a következő: Hogyan módosítsuk a modell paramétereit úgy, hogy a megfigyeléssorozat ezen paraméterekre vonatkozó feltételes valószínűsége, a $P(O|\lambda)$ maximális legyen?

Nem ismerünk analitikus megoldást erre a problémára. Nincs olyan módszer, amely tetszőleges rögzített megfigyeléssorozatra optimálisan közelítené a modell paramétereit. Léteznek azonban lokálisan konvergens eljárások, mint a Baum-Welch (BW) algoritmus [5], [6] , vagy a vele ekvivalens EM algoritmus [9], ill. gradiens módszerek [22]. Ebben a fejezetben egy, a BW algoritmuson alapuló iteratív eljárást mutatunk be a modell paramétereinek közelítésére.

Vezessük be a következő jelölést:

$$\xi_t(i,j) = P(q_t = S_i, q_{t+1} = S_j | O, \lambda).$$

A korábban használt $\alpha_t(i)$ és $\beta_t(j)$ változókkal kifejezhetjük $\xi_t(i,j)$-t:

$$\begin{aligned}\xi_t(i,j) &= \frac{P(q_t = S_i, q_{t+1} = S_j, O|\lambda)}{P(O|\lambda)} \\ &= \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}\end{aligned}$$

Emlékeztetőül, $\gamma_t(i)$-vel jelöltük annak a valószínűségét, hogy a t időpontban S_i állapotban vagyunk feltéve, hogy ismerjük az O megfigyeléssorozatot. $\gamma_t(i)$ is kifejezhető $\xi_t(i,j)$-vel:

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$$

Ha a $\gamma_t(i)$ értékeket t szerint összegezzük, akkor az O generálásakor bejárt állapotsorozatban az S_i előfordulásainak várható értékét kapjuk, vagy ami ezzel ekvivalens, az S_i állapotból induló állapotátmenetek várható értékét. (Utóbbi esetben csak $(T-1)$-ig összegzünk.)

Hasonlóan, a $\xi_t(i,j)$-k $t = 1, \ldots, T-1$ szerinti összege az $S_i \to S_j$ állapotátmenetek várható értékével egyezik meg. Összefoglalva:

$$\sum_{t=1}^{T-1} \gamma_t(i) = S_i\text{-ből induló állapotátmenetek várható értéke}$$

$$\sum_{t=1}^{T-1} \xi_t(i,j) = S_i \to S_j \text{ állapotátmenetek várható értéke}$$

Az eddigiek alapján már megadhatjuk a HMM paramétereire az iterációs formulákat:

$$\begin{aligned}
\bar{\pi}_i &= \text{az } S_i \text{ állapot előfordulásának várható értéke a } (t=1) \text{ pillanatban} \\
&= \gamma_1(i) \\
\bar{a}_{ij} &= \frac{S_i \to S_j \text{ állapotátmenetek várható értéke}}{S_i\text{-ből induló állapotátmenetek várható értéke}} \\
&= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \\
\bar{b}_j(k) &= \frac{\text{"}S_j \text{ állapotban } v_k\text{-t figyeltük meg" esetek előfordulásának várható értéke}}{\text{az } S_j \text{ állapot előfordulásának várható értéke}} \\
&= \frac{\sum_{\substack{1 \le t \le T \\ O_t = v_k}} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}
\end{aligned}$$

Jelölje a kezdeti modell paramétereit $\lambda = (A, B, \pi)$, a fenti egyenletekkel kapott új modell paramétereit $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$. Bizonyítható [6] , hogy vagy

- a kezdeti λ a likelihood függvény szélsőértéke, és ekkor $\bar{\lambda} = \lambda$; vagy

- a $\bar{\lambda}$ paraméterű modell jobban illeszkedik az O megfigyelés-sorozathoz, azaz $P(O|\bar{\lambda}) > P(O|\lambda)$.

Rögzített O megfigyeléssorozatból és véletlen λ paraméterű modellből kiindulva a fenti egyenletek alkalmazásával λ a likelihood függvény lokális maximumához konvergál, azaz a HMM rátanítható az O megfigyeléssorozatra.

5.2.5. Kiterjesztés folytonos esetre

Eddig olyan HMM-eket vizsgáltunk, amelyekben a megfigyelések egy véges ábécé szimbólumai voltak. A továbbiakban megfigyeléseink folytonos értékűek lesznek.

Ahhoz, hogy ilyen HMM-ekhez tanító algoritmusokat adhassunk, megszorításokat kell bevezetni a megfigyelések sűrűségfüggvényeire.

A legáltalánosabb sűrűségfüggvény, amelyhez már adtak HMM tanító algoritmust [23], [20], [21] a következő alakú:

$$b_j(\mathbf{O}) = \sum_{m=1}^{M} c_{jm} \Re[\mathbf{O}, \boldsymbol{\mu}_{jm}, \mathbf{U}_{jm}], \quad 1 \leq j \leq N \tag{5.6}$$

, ahol $\mathbf{O}$ a modellezni kívánt vektor, c_{jm} a komponens eloszlásokat "összekeverő" együtthatók, a $\Re$ pedig valamilyen log-konkáv[2] [23] (pl. Gauss) sűrűségfüggvény $\boldsymbol{\mu}_{jm}$ várható értékkel és $\mathbf{U}_{jm}$ kovarianciamátrixszal. A (5.6) alak segítségével tehát tetszőleges korlátos folytonos sűrűségfüggvényt M darab log-konkáv komponens lineáris kombinációjával közelíthetjük. A gyakorlatban $\Re$ Gauss eloszlás sűrűségfüggvénye szokott lenni. A c_{jm} együtthatókra igazak a

$$\begin{aligned} \sum_{m=1}^{M} c_{jm} &= 1, & 1 \leq j \leq N \\ c_{jm} &\geq 0, & 1 \leq j \leq N \\ & & 1 \leq m \leq M \end{aligned}$$

megszorítások, így $b_j(\mathbf{O})$ valóban sűrűségfüggvény, ui. nemnegatív és

$$\int_{-\infty}^{\infty} b_j(x)dx = 1, \quad 1 \leq j \leq N.$$

Bizonyítható [23], [20], [21] , hogy a $c_{jk}, \boldsymbol{\mu}_{jk}, \mathbf{U}_{jk}$ paraméterek megfelelő iterációs formulái a következők:

[2] Az $f(x)$ log-konkáv, ha $\ln(f(x))$ konkáv függvény.

$$\bar{c}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T} \sum_{k=1}^{M} \gamma_t(j,k)}, \tag{5.7}$$

$$\bar{\boldsymbol{\mu}}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k) \cdot \mathbf{O}_t}{\sum_{t=1}^{T} \gamma_t(j,k)}, \tag{5.8}$$

$$\bar{\mathbf{U}}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k) \cdot (\mathbf{O}_t - \boldsymbol{\mu}_{jk})(\mathbf{O}_t - \boldsymbol{\mu}_{jk})'}{\sum_{t=1}^{T} \gamma_t(j,k)}, \tag{5.9}$$

ahol a vessző transzponálást jelöl, a $\gamma_t(j,k)$ pedig annak valószínűségét, hogy a t pillanatban S_j állapotban vagyunk feltéve, hogy ismerjük az O megfigyelést, és beleszámítva az O_t k-adik komponens szerinti valószínűségét, azaz

$$\begin{aligned}
\gamma_t(j,k) &= \left[\frac{\alpha_t(j)\beta_t(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)} \right] \left[\frac{c_{jk}\Re(\mathbf{O}_t, \boldsymbol{\mu}_{jk}, \mathbf{U}_{jk})}{\sum_{m=1}^{M} c_{jm}\Re(\mathbf{O}_t, \boldsymbol{\mu}_{jm}, \mathbf{U}_{jm})} \right] \\
&= \gamma_t(j) \left[\frac{c_{jk}\Re(\mathbf{O}_t, \boldsymbol{\mu}_{jk}, \mathbf{U}_{jk})}{\sum_{m=1}^{M} c_{jm}\Re(\mathbf{O}_t, \boldsymbol{\mu}_{jm}, \mathbf{U}_{jm})} \right].
\end{aligned}$$

A $\gamma_t(j,k)$ tehát a $\gamma_t(j)$ speciális, az összetett sűrűségfüggvény egy komponensére vonatkozó változata. Értékük $M = 1$ esetben egyezik meg, vagyis ha a "keverék" egyetlen függvényből áll.

5.3. Alternatív modellek

Ebben a pontban megvizsgáljuk a HMM-ek gyenge pontjait a beszéd modellezésében, majd említés szintjén foglalkozunk más modellezési lehetőségekkel.

A Rejtett Markov Modellek a legszélesebb körben alkalmazott akusztikus modellek a mai beszédfelismerőkben, holott két fontos feltevéssel élnek, melyek nem teljesülnek a beszédre [14],[17]:

- A beszéd diszkrét állapotokra bontható, amely állapotokban a generált jel stacionárius és ezen állapotok között az átmenet azonnali.
- Annak a valószínűsége, hogy egy megfigyelési érték egy adott állapothoz tartozik, csak a megfigyelési értéktől és az állapottól függ, azaz nem függ az őt megelőző és követő megfigyelési értékektől.

HMM-ek esetén az első problémára az modell állapotszámának növelése lehet megoldás, azonban így jelentősen megnő a modell paramétereinek száma, ami nagyban növeli a tanító algoritmus futási idejét. Az emberi beszédgeneráló szervek fizikai tulajdonságai miatt a beszédre a második feltevés sem teljesül. Ezt a HMM-ek esetén úgy orvosolják, hogy az előfeldolgozó által generált megfigyelési értékekbe időre vonatkozó információt csempésznek az első és másodrendű delta együtthatókkal, holott ez ellentmond a Markov feltevésnek, miszerint azok időben egymástól függetlenek kivéve a közvetlen megelőzőt.

A Rejtett Markov Modellek ezen hiányosságainak pótlására alkalmazzák az *állapottér model*leket. Ezek funkiconalításukat tekintve két részre bonthatók. Egyrészről tartalmazzák a beszédgenerálás belső állapotainak valamilyen reprezentációját, ezt úgy képzelhetjük el, mint például az artikulátorok pozícióit és azok időbeli változását, ezt nevezik *állapot fejlődés folyamat*nak (state evolution process). Másrészt az ezen belső állapotok és nekik megfelelő megfigyelések közti leképezést megvalósító, *megfigyelés hozzárendelés*t (observation mapping) [32].

A legszembetűnőbb különbség a Rejtett Markov Modellek és az állapottér modellek között, hogy a HMM-eknél látott diszkrét állapotokkal szemben az aktuális állapotot itt folytonos érték írja le. Az állapottér modell általánosan a k-dimenziós x_t állapot vektorral és a p dimenziós o_t megfigyelési értékkel írható le, ahol $k < p$ és a következő generatív összefüggés érvényes:

$$\begin{aligned} x_{t+1} &= f(x_1, x_2, \ldots, x_t, w_t) \\ o_t &= g(x_t, v_t) \end{aligned} \tag{5.10}$$

ahol $f(.)$ az állapot fejlődés folyamatát leíró függvény, míg $g(.)$ a megfigyelés hozzárendelést megvalósító leképezés. w_t és v_t az ú.n. állapot fejlődési zaj és megfigyelés hozzárendelési zaj a rendszerben. Bár az általános keret semmilyen megkötést nem tesz az $f(.)$ és a $g(.)$ függvények linearitására vagy nem-linearitására, a továbbiakban csak lineáris esetre látunk majd példát. A lineáris Gauss modellek definíciója alapján a w_t és v_t a következő Gauss eloszlásokból jönnek:

$$\begin{aligned} w_t &\sim \mathcal{N}(\mu_t^{(x)}, \Sigma_t^{(x)}) \\ v_t &\sim \mathcal{N}(\mu_t^{(0)}, \Sigma_t^{(0)}) \end{aligned} \tag{5.11}$$

A $g(.)$ függvény szerepe úgy képzelhető el, mint az artikulátorok aktuális pozíciójának, azaz x_t-nek leképezése az adott pillanatban kiadott hangra, o_t-re amelyet megzavar v_t, mint a környezetből származó zaj.

5.3.1. Állapot fejlődés folyamat

Két speciális állapot fejlődés folyamatot mutatunk most be. Az egyiket félig-konstans állapot fejlődésnek nevezzük és az alapjául egy Rejtett Markov Modell szolgál, míg a második a lineáris folytonos állapot fejlődés.

A félig-konstans állapot fejlődés állapot vektorait egy Rejtett Markov Modell generálja. Láttuk, hogy a HMM-et az állapotátmeneti valószínűségek a_{ij} és az egyes állapotokhoz tartozó megfigyelés eloszlások $b_j(o_t)$ jellemezik. Ebben az esetben a HMM által generált megfigyelések a modell folytonos állapotai lesznek. A folytonos állapotvektorok mindaddig ugyanabból a HMM diszkrét állapotához tartozó eloszlásból jönnek, míg a HMM állapotot nem vált, innen a név félig-konstans állapot fejlődés. Az ilyen rendszerek állapot fejlődését a következő összefüggés jellemzi:[3]

$$\begin{aligned} q_t &\sim P(q_t|q_{t-1}) \\ x_t &\sim w_{q_t} \\ w_j &\sim \sum_n c_{jn}^{(x)} \mathcal{N}(\mu_{jn}^{(x)}, \Sigma_{jn}^{(x)}) \end{aligned} \tag{5.12}$$

[3]A $\sim$ szimbólum az első sorban a Markov tulajdonságot szimbolizálja.

A lineáris folytonos állapot fejlődés esetén továbbra is érvényes Markov feltevés, azaz a t. időpillanatban x_t közvetlenül csak x_{t-1}-től függ, közöttük azonban most egy lineáris leképezés érvényes. Az állapot fejlődést továbbá megzavarja egy Gauss eloszlású w zaj. Az állapot fejlődést a következő összefüggés jellemzi:

$$\begin{aligned} x_{t+1} &= Ax_t + w \\ w &\sim \mathcal{N}(\mu^{(x)}, \Sigma^{(x)}) \end{aligned} \tag{5.13}$$

ahol A egy k szor k-s állapot fejlődés mátrix és w a fejlődési zaj.[4]

5.3.2. Megfigyelés hozzárendelés

A megfigyelés hozzárendelés egy lehetséges módja, amit az irodalomban faktor analízisnek is neveznek a következő:

$$\begin{aligned} o_t &= Cx_t + v \\ v &\sim \mathcal{N}(\mu^{(0)}, \Sigma^{(0)}) \end{aligned} \tag{5.14}$$

ahol C egy p x k-s megfigyelési mátrix, v pedig a megfigyelési zaj. A megfigyelési zaj értéke független az adott időpillanat x_t állapotvektorától és általában feltétel, hogy diagonális kovariancia mátrix-al rendelkezik.

5.4. Lineáris Dinamikus Rendszerek (LDS)

Az előzőek alapján tekintsünk egy olyan modellt, amelyet lineáris folytonos állapot fejlődés és faktor analízis jellemez, az ilyen rendszert hívjuk lineáris dinamikus rendszernek és a következő képpen írható le:

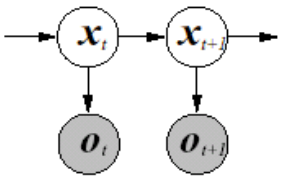
$$\begin{aligned} x_t &= Ax_{t-1} + w \\ w &\sim \mathcal{N}(\mu^{(x)}, \Sigma^{(x)}) \\ o_t &= Cx_t + v \\ v &\sim \mathcal{N}(\mu^{(0)}, \Sigma^{(0)}) \end{aligned} \tag{5.15}$$

ahol w és v független Gauss eloszlású valószínűségi változók diagonális kovariancia mátrix-al és a kezdeti állapot a következő eloszlásból jön:

[4] Az A mátrix nem összekeverendő a HMM-eknél ugyancsak A-val jelölt átmeneti mátrix-al.

$$x_1 \sim \mathcal{N}(\mu^{(1)}, \Sigma^{(1)})$$

A lineáris dinamikus rendszernek $(4 + k)k + (2 + k)p$ paramétere van, ahol a kezdeti állapot és az állapot fejlődési zajnak összesen $4k$, az állapot fejlődési mátrixnak k^2, a megfigyelési mátrixnak pk és a megfigyelési zajnak $2p$ paramétere van. Ezen paraméterek hangolására szolgáló algoritmust részleteit lásd a [32, 11]-ban. A 5.6. ábra az LDS Bayes-hálóját szemlélteti.



5.6. ábra. **Lineáris Dinamikus Rendszer szemléltetése Bayes-hálóval.**

6. fejezet

Hang-felismerés képfeldolgozással

Az ELTE NIPG csoportjában fejlesztett szemdetektor [25] a Viola és Jones [42] által publikált technológiára épül, amely egy robusztus valós-idejű objektum detektor. Ez a technológia alapvetően a képfeldolgozás területén használt, a szerzők hatékony arcdetektor rendszer fejlesztésére tervezték. Ebben a fejezetben ezen technológia alkalmazását mutatjuk be a hangfelismerés területén. Detektorokat fejlesztettünk az Intel OpenCV[1] könyvtár segítségével, melyek a hang spektrális képében keresnek bizonyos hangok képeinek megfelelő objektumokat. Ez a módszer az általános keretben az egyszerű minta-felismerő rendszereknek felel meg, amit a 3.1. ábra szemléltet. A továbbiakban ismertetjük a Viola-Jones detektor rendszer elméleti hátterét (6.1. pont), majd az általunk használt adatbázist és a teszteredményeket. A fejezet részben Kiszlinger Melinda [25] dolgozata alapján készült.

6.1. Viola-Jones objektum detektálás

Ebben a pontban bemutatjuk a Paul Viola és Michael J. Jones által fejlesztett robusztus valós-idejű objektum detektort [42]. A rendszer robusztussága a képfeldolgozás sebességében és a magas találati arányban rejlik. A technológiára támaszkodva kialakítottak egy frontális arcdetektor rendszert, mely eléri az előtte publikált legjobb eredmények [39, 34, 36, 33, 2] *találati és hamis pozitív arányát.* Más arcdetektáló rendszerekben, kisegítő információkat használnak, hogy magas képfeldolgozási arányt érjenek el. Ilyen például egymás utáni képek különbsége videó filmeken vagy pixel-szín színes képeken.

[1] http://sourceforge.net/projects/opencvlibrary/

A Viola-Jones detektor szürkeskálás képekből kapott információk alapján képes magas képfeldolgozási arányt elérni. Az említett egyéb információ források is integrálhatók a Viola-Jones rendszerbe, így még magasabb detektálási arányt érhetünk el. A rendszer részleteinek tisztázásához bevezetjük a következő fogalmakat.

- **Hamis találat vagy hamis pozitív**: Ha a detektor a kép egy részletén igaz eredménnyel tér vissza, miközben a képrészleten nem szerepel a keresett objektum, akkor azt hamis találatnak vagy hamis pozitívnak nevezzük.
- **Hamis negatív**: Ha a detektor a kép egy részletén nem jelez találatot, miközben a képrészletén szerepel a keresett objektum, akkor azt hamis negatívnak nevezzük.
- **Találat**: Ha a detektor a kép egy részletre igaz eredményt ad, és a képrészlet a keresett objektumot valóban ábrázolja, akkor ezt az eseményt találatnak nevezzük.
- **Hamis pozitív arány vagy hamis találati arány**: Detektor tesztelése során a hamis találatok számát osztva a vizsgált képrészletek számával kapjuk a hamis pozitív arányt vagy hamis találati arányt. Ez egy valós szám a $[0..1]$ intervallumból.
- **Találati vagy detektálási arány**: Detektor tesztelése során a találatok számát osztva a keresett objektumok számával kapjuk a találati vagy detektálási arányt. Ez egy valós szám a $[0..1]$ intervallumból.

A következő fontos tulajdonságai vannak a Viola-Jones rendszernek, amit a későbbiekben részletesen ismertetünk:

1. Alternatív képreprezentáció, melyet *integrális képnek* nevezünk. Az integrális kép segítségével nagyon gyors *jellemző* (feature) kiértékelés válik elérhetővé. Részben Papageorgiou és társai munkája [28] által motiváltan a Viola-Jones rendszer nem közvetlenül a képintenzitásokkal dolgozik, hanem képrégiók jellemzőivel. Hasonlóan Papageorgiou-hoz, jellemzők egy halmazát használja, melyek hasonlóak a Haar-féle bázisfüggvényekhez. Az integrális kép képpontonként néhány elemi művelettel számítható ki a képből. Az integrális kép ismeretében pedig a Haar jellemzők konstans időben számolhatók minden skálán és pozíción.
2. Az AdaBoost tanuló algoritmus alapján osztályozó létrehozása a fontos jellemzők egy kis halmazának kiválasztásával.[13]. Egy

kép részablakán belül az összes Haar jellemzők száma nagyon nagy, sokkal nagyobb, mint a pixelek száma. Azért, hogy gyors osztályozást biztosítsunk, a tanuló folyamatnak ki kell zárnia az elérhető jellemzők nagy többségét, és a kritikus jellemzők kis halmazára kell koncentrálnia. Tieu és Viola munkája által motiváltan, a jellemző kiválasztás az AdaBoost egy egyszerű módosítása: minden gyenge osztályozót kényszerítenek, hogy annak eredménye csak egy jellemzőtől függjön [40]. Ennek eredményeként a Boost folyamat minden köre, amely új gyenge osztályozót választ, megfelel egyetlen jellemzőt kiválasztó folyamatnak [35, 27, 28].

3. Több bonyolultabb osztályozó egymás után kombinálása, *kaszkád struktúrába, sorba rendezése*, mely amellett, hogy lényegesen megnöveli a detektor sebességét a kép sokat ígérő területeire fokuszál. Egy képen gyakran gyorsan meghatározható, hol fordulhat elő egy objektum [41, 19, 2]. Még összetettebb folyamatokat a sokat ígérő részek számára tartunk fenn. Legfőbb mértéke egy ilyen megközelítésnek a vizsgálati folyamat hamis negatív aránya, azaz a nem detektált objektumok aránya. Ezért a figyelmi szűrőnek mindegyik vagy majdnem mindegyik objektumot ki kell választania.

Láthatjuk majd egy nagyon egyszerű és hatékony osztályozó tanulási folyamatát, amely "felügyelt" figyelmi szűrőként használható. A felügyelt kifejezés arra a tényre utal, hogy a figyelmi operátort úgy tanítjuk, hogy sajátos osztályok példányait detektálja. Az arcdetektálás terén lehetséges, hogy kevesebb, mint 1% hamis negatív arányt, és 40% hamis pozitív arányt érjen el egyetlen osztályozó, figyelmi szűrő, mely 20 egyszerű művelettel kiértékelhető. Ennek a szűrőnek a hatása, hogy kevesebb, mint felére csökken azon területek száma, ahol a végső detektort kiértékelésre kerül.

Azok a kép-részletek, melyeket a kezdeti osztályozó, vagyis a figyelmi szűrő nem utasít el, feldolgozásra kerülnek az osztályozók szekvenciájában, melynek minden egyede valamivel komplexebb az előzőnél. Ha az adott részletet bármelyik osztályozó elutasítja, az kiesik a feldolgozási folyamatból. A kaszkád detektálási folyamat lényegében egy degenerált döntési fa, és mint ilyen, kapcsolatban áll Amit és Geman munkájával [2].

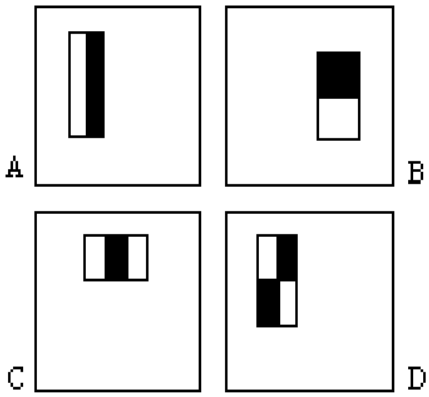
6.1.1. Jellemzők (features)

A Viola-Jones objektum detektáló rendszer egyszerű jellemzők értékei alapján osztályoz. Számtalan érv szól a jellemzők használata

mellett, a képpontok közvetlen használata helyett. A legáltalánosabb ok, hogy a jellemzők képesek olyan tudás területeket kódolni, melyeket nehéz megtanulni véges mennyiségű tanító adathalmazt használva. A rendszer egy másik fontos motivációja a jellemzők használatára a sebesség. A jellemző-alapú rendszer lényegesen gyorsabban, mint a képpont-alapú.

A használt egyszerű jellemzők hasonlítanak a Haar-féle bázisfüggvényekre, melyeket Papageorgiou és társai használtak [28]. A Viola-Jones rendszer ezeknek a jellemzőknek három fajtáját használja. A *két-téglalap jellemző* értéke a pixelek összegének különbsége két téglalap terület között. A területeknek ugyanakkora a mérete és alakja, és függőlegesen vagy vízszintesen szomszédosak (6.1. ábra). A *három-téglalap jellemző* két szélső téglalap pixeleinek összegét vonja le a középső téglalap pixeleinek összegéből. Végül a *négy-téglalap jellemző* diagonális téglalap párok közötti különbséget számít.

Feltételezve, hogy a detektor alap felbontása 24x24-es, a téglalap jellemzőknek halmaza 45396 elemű.



6.1. ábra. **Viola-Jones téglalap jellemzők (features).** A fekete és a fehér téglalapon belüli pixelek összegének különbsége adja a jellemző értékét. A és B a két-téglalap, C a három-téglalap, D pedig a négy-téglalap jellemzőt szemlélteti.

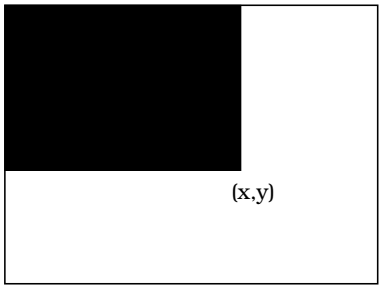
6.1.2. Integrális kép

A téglalap jellemzőket gyorsan ki lehet számolni a kép egy alternatív reprezentációját használva, melyet integrális képnek nevezünk. Az integrális kép az x, y pontban a tőle balra fent elhe-

lyezkedő pixelek intenzitásértékeinek összege:

$$ii(x,y) = \sum_{x' \leq x, y' \leq y} i(x',y') \tag{6.1}$$

ahol $ii(x,y)$ az integrális kép, és $i(x,y)$ az eredeti kép (6.2. ábra).



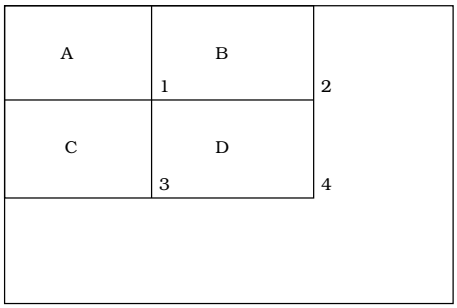
6.2. ábra. **Integrális kép.** Az integrális kép értéke az (x,y) pontban a tőle balra fent elhelyezkedő pixelek intenzitásértékeinek összege.

Használjuk a következő rekurzív párokat:

$$s(x,y) = s(x,y-1) + i(x,y) \tag{6.2}$$

$$ii(x,y) = ii(x-1,y) + s(x,y) \tag{6.3}$$

Ahol $s(x,y)$ az intenzitás értékek halmozódó sorösszege, $s(x,-1) = 0$, és $ii(-1,y) = 0$. Az integrális képet így egyszer kell csak számolni az eredeti képből, majd a későbbiekben a jellemző kiértékelésben használjuk.



6.3. ábra. **Az integrális kép előnye a téglalap összeg számítása esetén.**

Az integrális képet használva egy téglalap összeg négy tömbhivatkozással számítható (6.3. ábra). A D-be eső képpontok összege

az integrális kép négyszeri elérésével meghatározható. Az integrális kép értéke az 1-es pozícióban az A-ba eső képpontok intenzitásösszege. 2-ben az érték A + B, 3-ban A + C, 4-ben A + B + C + D. Így a D-be eső összeg a 4 + 1 - (2 + 3) összefüggés alapján számítható ki.

Két téglalap közötti különbség nyolc hivatkozással számítható. A két-téglalap jellemző mivel szomszédos téglalapok különbségét veszi, hat tömbhivatkozással számítható, a három-téglalap jellemző nyolc hivatkozással, a négy-téglalap jellemző kilenc hivatkozással.

A téglalap jellemzők valamivel egyszerűbbek összehasonlítva más alternatívákkal, mint például a steerable filter-ek [12, 15]. A steerable filter-ek és rokonaik kitűnően alkalmazhatók a határok részletezett elemzésére, képtömörítésre, és textúra elemzésre. Ezzel szemben a téglalap jellemzők egészen durvák, habár érzékenyek az élek, vonalak és más egyszerű struktúrák jelenlétére. Eltérően a steerable filter-ektől a téglalap jellemzőkkel csak a horizontális és vertikális irányítások elérhetők. Mégis a téglalap jellemzők halmaza gazdag képreprezentációt nyújt, ami segíti a hatékony tanulást és a magas számítási hatékonyság kompenzálja a korlátozott rugalmasságot.

Vizsgáljunk meg egy hagyományos szemléletet, amiben a kép piramisát számoljuk. Hasonlóan a legtöbb objektum detektáló rendszerhez a Viola-Jones detektor több skálán is végignézi a képet; az alap helyzetből indul, ahol az objektumokat 24x24 pixelben keresi, majd a képet 11 skálán tapogatja le, mindegyik skála növekedési faktora 1.25. A hagyományos megközelítés 11 piramiskép kiszámítása, ahol mindegyik kép 1.25-ször kisebb az előzőnél. Ezután egy fix méretű detektor tapogatja le a képeket. A piramis képek kiszámítása jelentős időt igényel. Hagyományos hardware használatával igen nehéz 15 kép/másodperc mellett kiszámolni a piramis képeket[2].

Ezzel szemben Viola és Jones definiálták a jellemzők egy jelentős halmazát, amelyek tulajdonában egy egyszerű jellemző bármely skálán és pozícióban kevés művelettel számítható. A 6.1.4. fejezetben ismertetjük, hogyan hozható létre hatékony detektor akár néhány két-téglalap jellemző segítségével. A jellemzők kiszámításának hatékonysága miatt az arcdetektor végigvihető egy teljes képen minden skálán és pozícióban 15 kép/másodperc mellett, kevesebb idő alatt, mint ami a 11 piramis kép kiszámításához kell. Ezért bármely detektor, amihez ki kell számolni a képek piramisát, lassabban fut a Viola-Jones arcdetektornál.

[2]A képpontok száma 11 szinten körülbelül $55 * 384 * 288 = 6082560$. Feltéve, hogy minden pixel 10 műveletet igényel a kiszámításhoz, a piramis körülbelül 60,000,000 műveletbe kerül. Tehát körülbelül 900,000,000 művelet/másodperc kell a 15 kép/másodperces feldolgozáshoz.

6.1.3. Osztályozó függvények tanítása, Adaboost

Amennyiben adott a jellemzők, továbbá pozitív és negatív tanítópéldák egy-egy halmaza, a gépi tanulás bármely megközelítése használható egy osztályozó függvény tanulásához. Sung és Poggio a Gauss modell egy keverékét használják [39]. Rowley, Baluja és Kanada egyszerű jellemzők egy kis halmazát és neuronhálót használnak [34]. Osuna és mások SVM-et (support vector machine-t) használnak [27]. Roth és társai egy új és eddig nem használt képreprezentációt javasoltak és használtak a Winnow tanulási folyamattal együtt [33].

Említettük, hogy egy kép részletnek 45,396 téglalap jellemzője van. Ez a szám sokkal nagyobb, mint a képpontok száma. Habár minden jellemző hatékonyan számítható, a jellemzők teljes halmazának számítása megengedhetetlen. A jellemzők igen kicsi halmazával is létrehozható hatékony osztályozó. Az igazi kihívás ezen az osztályozók a megtalálása.

A Viola-Jones rendszerben az AdaBoost egy változatát használják mind a jellemző kiválasztásra, mind pedig az osztályozó tanítására. Eredeti formájában az AdaBoost gyenge osztályozó függvények kombinálásával képes egy erős osztályozót előállítani, ezáltal tetszőleges tanuló algoritmus javítására használható. Az alapgondolat a következő: a gyenge osztályozót lefuttatjuk egy tanítóhalmazon. A teljes tanítóhalmaz osztályozása után az előző osztályozó által hibásan osztályozott példákat újra súlyozzuk, és újra elvégezzük a tanítást. Így a végső erős osztályozó egy perceptron lesz, a gyenge osztályozók lineáris kombinációja, kiegészítve egy küszöbértékkel.

Freund és Shapire bizonyította, hogy az erős osztályozó tanulási hibája a körök számával exponenciálisan tart nullához [35].

A hagyományos AdaBoost módszert egy mohó jellemző kiválasztó folyamatként értelmezhető. A jellemző osztályozó függvények egy nagy halmazát kombináljuk össze súlyozott többségi szavazás alapján. Az igazi kihívás, hogy a jó osztályozókhoz nagy súlyt adjunk, míg a kevésbé jó függvényekhez kicsit. Az AdaBoost egy agresszív eljárás jó osztályozók kis halmazának kiválasztására, amik jelentősen eltérőek lehetnek.

Egy gyakorlati eljárás ennek az analógiának a kiegészítésére, hogy megszorítjuk a gyenge tanulót az osztályozó függvényeknek arra a halmazára, melyek csak egyszerű jellemzőktől függnek. Ennek eléréséhez a gyenge tanuló algoritmust úgy tervezték meg, hogy egy egyszerű téglalap jellemzőt válasszon, ami a legjobban szétválasztja a pozitív és negatív példákat (ez hasonló [40] megközelítéséhez a képvisszakeresés területén). Minden jellemzőhöz a gyenge tanuló meghatározza az optimális küszöb osztályozó függvényt, ami a példák minimális számú halmazát osztályozza rosszul. Így egy gyenge osztályozó

$(h_j(x))$ egy jellemzőből (f_j), egy küszöbből (Θ_j) és egy paritásból (p_j) áll, ahol a paritás az egyenlőtlenség jel irányát jelöli:

$$h_j(x) = \begin{cases} 1 & \text{ha } p_j f_j(x) < p_j \Theta_j \\ 0 & \text{különben} \end{cases}$$

ahol x egy 24x24 pixel felbontású képrészlet.

Gyakorlatban nem lehet egyszerű jellemzővel megvalósítani az osztályozást úgy, hogy alacsony hibát érjünk el. A folyamat korai szakaszában kiválasztott jellemzők 0.1 és 0.3 közötti hibaarányt adnak. A későbbi körökben választott jellemzők, melyek egyre bonyolultabbak, 0.4 és 0.5 közötti hibaarányt adnak.

A tanulóalgoritmus, mely egy kérdés online tanulásának menetét írja le a következő. Minden T hipotézis egy egyszerű jellemzőt használ. A végső hipotézis a T hipotézisek egy súlyozott lineáris kombinációja, ahol a súlyok fordítottan arányosak a tanulási hibával.

- Adottak n darab tanítóminta $(x_i, y_i), \ldots, (x_n, y_n)$, ahol $y_i = 0, 1$ a negatív illetve pozitív képeknél.
- A kezdeti súlyok $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ minden $y_i = 0, 1$ -nek megfelelően, ahol m és l a negatív illetve pozitív példák száma.
- Minden $t = 1, \ldots, T$-re (T darab gyenge osztályozót kombinálunk össze)

  1. Normalizáljuk a súlyokat úgy, hogy w_t a valószínűségi eloszlás legyen,
  $$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,i}}.$$
  2. Minden j jellemzőre tanítsunk egy h_j osztályozót, ami csak egyetlen egyszerű jellemzőt használhat. A w_t-vel súlyozott hiba $\epsilon_j = \sum_i w_i |h_j(x_j) - y_i|$.
  3. Válasszuk ki h_t osztályozót, melynek a legkisebb ϵ_t hibája van.
  4. Súlyozzuk át a tanítóhalmazt a következőképpen:
  $$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i},$$
  ahol $e_i = 0$ ha az x_i példát helyesen osztályoztuk, $e_i = 1$ különben, és $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
  5. A végső erős osztályozó az így kapott gyenge T darab h_t osztályozó kombinációja:
  $$h(x) = \begin{cases} 1 & \sum_{t-1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t-1}^{T} \alpha_t \\ 0 & \text{különben} \end{cases}$$
  ahol $\alpha_t = \log \frac{1}{\beta_t}$.

6.1.4. Osztályozók kaszkádba szervezése

Ebben a pontban ismertetjük a kaszkádba szervezés menetét, melynek hatására nő a detektálási arány és jelentősen csökken a számítás igény. Az alapötlet, hogy könnyebb olyan gyenge osztályozót készíteni, amely elutasítja a hamis képrészletek többségét, azzal szemben hogy minden pozitív példát detektálna. Egyszerű osztályozókat használunk tehát a negatív példák elutasítására, és az összetettebb osztályozókat csak a ígéretes területeken futtatjuk le, hogy minél alacsonyabb hamis pozitív arányt érjünk el.

A kaszkád szintjei az AdaBoostot használó osztályozók tanulása során jönnek létre. Kettő jellemzőből álló erős osztályozóval kezdve hatékony arcszűrőt kaphatunk az erős osztályozó küszöbének olyan beállításával, amely minimalizálja a hamis negatívok számát. Az AdaBoost kezdeti küszöbértéke, $\frac{1}{2}\sum_{t=1}^{T}\alpha_t$, a tanítóhalmazon minimalizálja a hibaarányt. Egy alacsonyabb küszöb magasabb detektálási arányt és magasabb hamis pozitív arányt eredményez. A mérésekre alapozva egy érvényes tanítóhalmazt használva, a kettő jellemzős osztályozó beállítható, hogy az arcok 100%-át detektálja 40% hamis pozitív eredmény mellett.

A kettő jellemzőből álló osztályozó detektálási teljesítménye a gyakorlatban még nem elfogadható. Mindamellett az osztályozó kevés művelettel jelentősen csökkenti azon képrészletek számát, melyek további feldolgozást igényelnek:

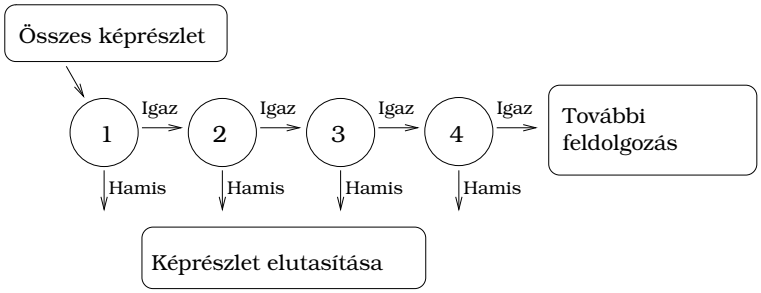
1. A téglalap jellemzők kiértékelése (jellemzőnként 6-9 tömbhivatkozás).

2. A gyenge osztályozó kiszámítása (minden jellemzőre egy műveletet, egy küszöbölés).

3. A gyenge osztályozók kombinálása (jellemzőnként egy szorzás, egy összeadás és végül egy küszöbölés).

A két jellemzős osztályozó körülbelül 60 processzor utasításba kerül. Összehasonlítva egy egyszerű kép letapogatással, vagy egy egyrétegű perceptronnal, a kettő jellemzős módszer 20-szor kevesebb időt vesz igénybe alablakonként.

Az általános formája a detektálási folyamatnak egy döntési fa, melyet kaszkádnak nevezünk [29]. Ez szemlélteti 6.4. ábra. Egy pozitív eredmény az első osztályozóból kiváltja a második osztályozó kiértékelését, amely szintén úgy van beállítva, hogy magas detektálási arányt érjen el. Egy pozitív eredmény a második osztályozóból kiváltja a harmadik osztályozó kiértékelését, és így tovább. Egy negatív eredmény bármely szinten az alablak azonnali elutasításához vezet.



6.4. ábra. **Detektor kaszkád.**

A kaszkád struktúra célja azon tény kihasználása, hogy egy kép legtöbb részlete negatív. Ezért a kaszkád minél több negatívot próbál elutasítani a legkorábbi szinteken. A pozitív egyed kiváltja az összes szint kiértékelését, ami ritka esemény.

Hasonlóan egy döntési fához, a részsorozat egy osztályozóját azokkal a példákkal képezzük ki, melyek az összes előző szakaszon átmentek. Ennek eredményeként a második osztályozó bonyolultabb mint az első. A példák melyek átmennek az első szinten, "nehezebbek" mint a tipikus példák. Adott detektálási arány mellett, a mélyebben lévő osztályozók hamis pozitív aránya megfelelően magasabb.

Kaszkád szervezés algoritmusa

A kaszkádba szervezés célja tehát minél magasabb detektálási arány és minél alacsonyabb műveletigény elérése. Az arcdetektálás terén a régebbi rendszerek 85 és 95% közötti detektálási arányt értek el 10^{-5} és 10^{-6} hamis pozitív aránnyal. A kaszkád szintjeinek száma és a szintek mérete elegendő nagy kellett legyen, hogy legalább hasonló detektálási teljesítményt érjenek el, úgy, hogy közben a számításigényt csökkentsék.

Tegyük fel, hogy adott az osztályozók kaszkádja, melynek hamis pozitív aránya legyen

$$F = \prod_{i=1}^{K} f_i,$$

ahol F a kaszkád hamis pozitív aránya, K az osztályozók száma, és f_i az i. osztályozó hamis pozitív aránya a példákon, melyek eljutnak hozzá. A detektálási arány

$$D = \prod_{i=1}^{K} d_i,$$

ahol D a kaszkád detektálási aránya, K az osztályozók száma, és d_i az i. osztályozó detektálási aránya a példákon, melyek eljutnak hozzá.

Adott hamis pozitív és detektálási arány eléréséhez a cél arányok a kaszkád folyamat minden szintjéhez meghatározhatók. Például elérhetünk 0.9 detektálási arányt egy 10 szintes osztályozóval, ha minden szint detektálási aránya 0.99, mivel $0.9 \approx 0.99^{10}$. Míg ennek a detektálási aránynak az elérése ijesztően hangozhat, addig jelentősen könnyít a dolgon, hogy minden szintnek csak körülbelül 30%-os hamis pozitív arányt kell elérnie $(0.30^{10} \approx 6 \times 10^{-6})$.

A kép pásztázása során kiértékelt jellemzők száma szükségszerűen egy valószínűségi folyamat függvénye. Bármely képrészlet addig halad a kaszkádon, egyszerre csak egy osztályozón át, míg el nem döntjük, hogy az negatív, vagy ritka körülmények között a részlet az összes teszten végighalad, így pozitív eredményt ad. A folyamat várt viselkedését meghatározza a kép ablakok szétosztása egy tipikus teszt halmazon. Az osztályozók egy kulcs-mértéke azok "pozitív aránya", azaz azoknak az ablakoknak az aránya, melyekre az osztályozó pozitív választ ad. A kiértékelt jellemzők számának várható értéke:

$$N = n_0 + \sum_{i=1}^{K} \left(n_i \prod_{j<i} p_j \right)$$

ahol N a kiértékelt jellemzők várható száma, K az osztályozók száma, p_i az i. osztályozó pozitív aránya, és n_i a jellemzők száma az i. osztályban. Attól fogva, hogy az objektumok nagyon ritkák, a "pozitív arány" egyenlő a hamis pozitív aránnyal.

Az AdaBoost algoritmust, ellenben a magas detektálási arányok elérésével, a hibák minimalizálására tervezték. A hibák elkerülése az AdaBoost által létrehozott perceptron küszöbének beállításától függ. Magas küszöb kevesebb hamis pozitív találat elérésére képes osztályozót eredményez, alacsonyabb detektálási aránnyal. Az alacsony küszöb több hamis pozitívat és magasabb detektálási arányt elérő osztályozót ad.

A teljes tanítási folyamat kétféle, egymásnak ellentmondó kritérium optimalizálását igényli. Egy osztályozó több jellemzővel magasabb detektálási arányt és alacsonyabb hamis pozitív arányt érhet el. Ugyanakkor egy több jellemzővel rendelkező osztályozó több számítási időt igényel. Bevezetjük a következő optimalizációs keretet, amiben

- az osztályozó szintek száma,
- a jellemzők száma, n_i, minden szinthez, és
- minden szint küszöbe

úgy kivitelezett, hogy minimalizálja N jellemzők számát, adott F és D célértékek mellett.

Gyakorlatilag egy egyszerű alkalmazást használnak hatékony osztályozók előállításához. A felhasználó kiválasztja a minimum elfogadható f_i és d_i arányokat. A kaszkád minden szintjét AdaBoosttal tanítják, hogy a jellemzők számát addig növelik, míg a célul kitűzött detektálási és hamis pozitív arányokat el nem érik. Az arányokat a detektor tesztelésével kapják meg egy érvényes halmazon. Amíg nem érik el a célul kitűzött hamis pozitív arányt, további szinteket adnak a detektorhoz. A negatív halmazt a következő szintek tanításához az aktuális detektor futtatása során kapott hamis pozitív eredmények adják. Az algoritmus a következő:

- Határozzuk meg az f értéket, a maximálisan elfogadható hamis pozitív arányt, és d értéket, a minimálisan elfogadható találati arányt a kaszkád minden szintjéhez.
- Határozzuk meg a teljes kaszkádra vonatkozó hamis pozitív arányt, F_{target}-et.
- P = a pozitív példák halmaza.
- N = a negatív példák halmaza.
- $F_0 = 1.0; D_0 = 1.0$
- $i = 0$
- amíg $F_i > F_{target}$
  - $i \leftarrow i + 1$
  - $n_i = 0; F_i = F_{i-1}$
    - $n_i \leftarrow n_i + 1$
    - Használjuk P és N halmazokat egy n_i jellemzős osztályozó tanítására az AdaBoost algoritmussal.
    - Értékeljük ki az aktuális kaszkád osztályozót az érvényességi halmazon az F_i és D_i meghatározásához.
    - Csökkentsük az i. osztályozó küszöbét, amíg az aktuális kaszkád osztályozó eléri a minimum $d \times D_{i-1}$ detektálási arányt (ez befolyásolja F_i értékét is).
  - $N \leftarrow \emptyset$
  - Ha $F_i > F_{target}$, akkor értékeljük ki az aktuális kaszkád detektort a nem-objektum képeken és tegyünk minden hamis detektálást az N halmazba.

6.2. Adatbázis

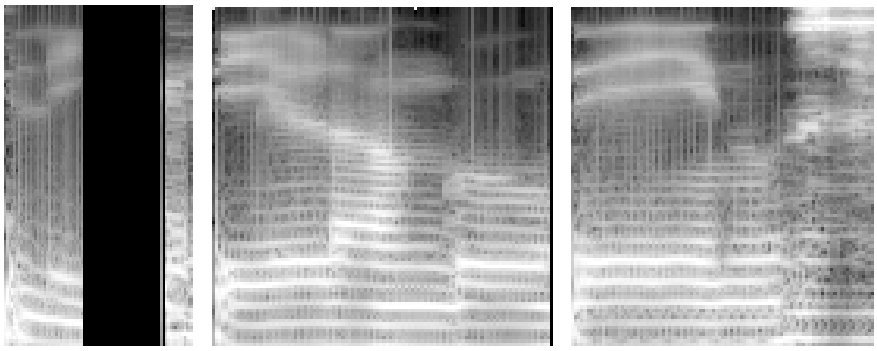
Az hangadatbázis saját felvételekből áll. Három különböző magyar szó a "klikk", a "vénusz" és a "pillangó", tételesen:

- 130 db "klikk".
- 150 db "vénusz".
- 130 db "pillangó".

A felvételek 5 különböző egyén hangját tartalmazzák, kettő különböző mikrofonnal készültek. 44.1kHz-es mono wav fájlokban lettek tárolva.

6.3. Tanítás és eredmények

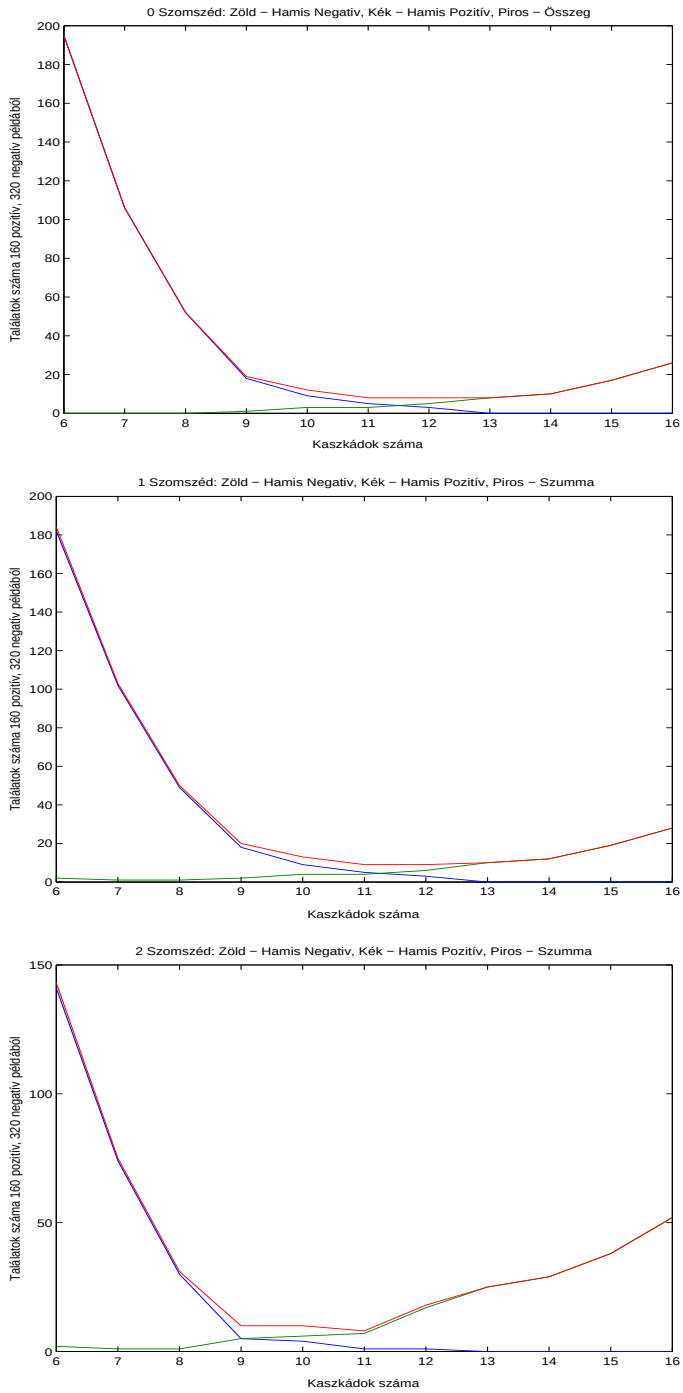
A tanítás során a szavak spektrális képeire készültek Viola-Jones detektorok. Az egyes szavak spektrális képeit szemlélteti a 6.5. ábra.



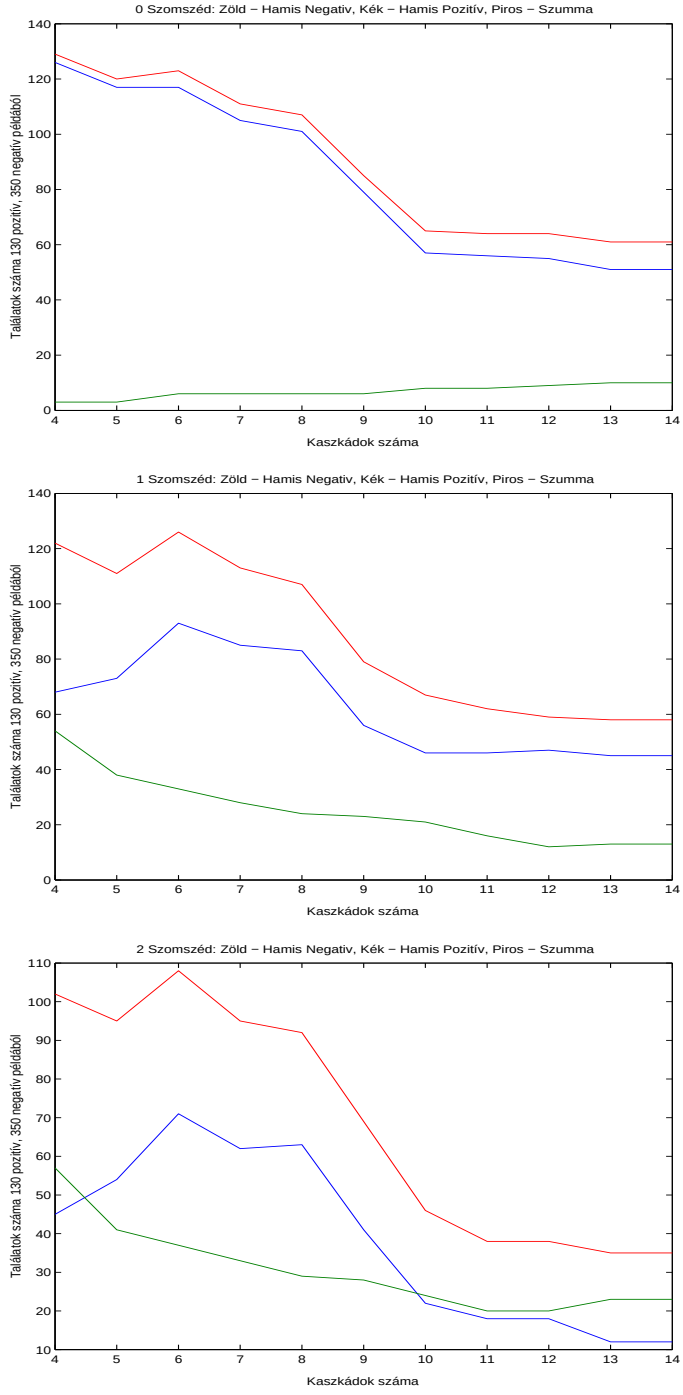
(a) "Klikk" **(b)** "Pillangó" **(c)** "Vénusz"

6.5. ábra. **A "klikk", a "pillangó" és a "vénusz" spektrális képe.**

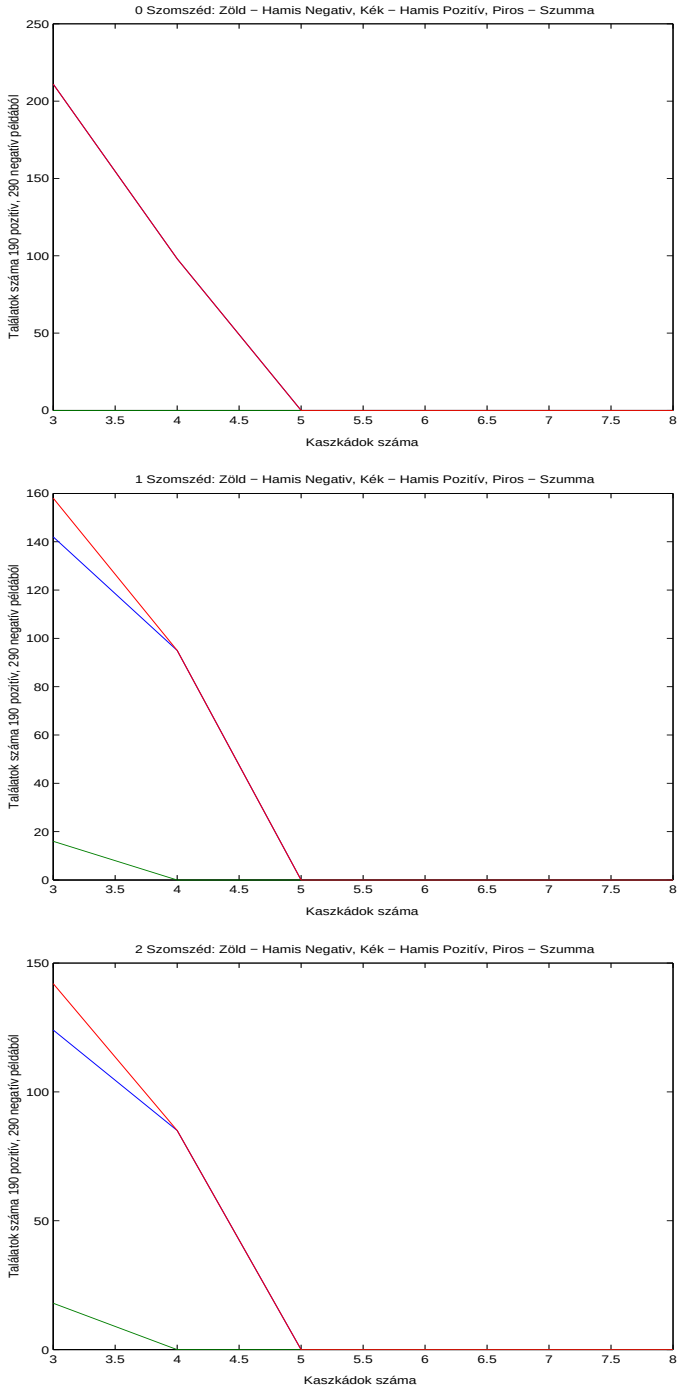
Az egyes szavakra tanított detektorok teszteredményeit mutatja a 6.6., 6.7., 6.8. ábra. Minden szóhoz három ábra tartozik, felülről lefelé a szükséges szomszédos talalátok száma alapján 0-tól 2-ig. A szomszédos találat azt jelenti, hogy a detektor által felismert régió akkor minősül felismerésnek, ha a detektor bizonyos távolsággal a találat körül további találatokat jelzett.



6.6. ábra. **A "klikk" szó felismerési eredményei.** Az x a kaszkádok, az y tengely a találatok száma. Felülről lefelé a 0, 1, illetve 2 szükséges szomszédos találat esete.



6.7. ábra. **A "pillangó" szó felismerési eredményei.** Az x a kaszkádok, az y tengely a találatok száma. Felülről lefelé a 0, 1, illetve 2 szükséges szomszédos találat esete.



6.8. ábra. **A "vénusz" szó felismerési eredményei.** Az x a kaszkádok, az y tengely a találatok száma. Felülről lefelé a 0, 1, illetve 2 szükséges szomszédos találat esete.

Ábrák jegyzéke

2.1. **Analóg jel szemléltetése folytonos függvénnyel.** . . 3
2.2. **Mintavételezés.** 5
2.3. **2kHz-es jel 10kHz-es mintavételezése.** 6
2.4. **4kHz-es jel 10kHz-es mintavételezése.** 6
2.5. **6kHz-es jel 10kHz-es mintavételezése.** 6
2.6. **Kvantálás.** 7

3.1. **Egyszerű minta kereső rendszer felépítése.** 9
3.2. **A beszédfelismerő rendszer felépítése.** 9

4.1. **Megfigyelési vektorok kivonása az audió jelből ablakozással.** 12
4.2. **A "vissza" szó Fourier transzformáltja.** 13
4.3. **Hamming ablak az idő és a frekvencia tartományban.** 14
4.4. **Mel-spektrum.** 16
4.5. **Mel Scale Filter Bank a lineáris spektrumban.** . . . 16
4.6. **Különböző frekvenciájú szinuszok szuperpozíciója.** 17
4.7. **Kürt hangjának spektruma visszhangos és tiszta környezetben.** 18
4.8. **Kürt hangjának kepsztruma visszhangos és tiszta környezetben.** 19
4.9. **A mel-scale kepsztrum számításának lépési.** 21

5.1. **Valószínűségi függőségek szemléltetése Bayes-hálókkal.** 25
5.2. **Rejtett Markov Modell szemléltetése Bayes-hálóval.** 26
5.3. **3 állapotú Markov modell.** 28
5.4. **Rejtett Markov Modell.**
Három rejtett állapottal rendelkező balról-jobbra modell megfigyeléseket generál. 30
5.5. **A Viterbi algoritmus szemléltetése.** 36

5.6. **Lineáris Dinamikus Rendszer szemléltetése Bayes-hálóval.** 43

6.1. **Viola-Jones téglalap jellemzők (features).** A fekete és a fehér téglalapon belüli pixelek összegének különbsége adja a jellemző értékét. A és B a két-téglalap, C a három-téglalap, D pedig a négy-téglalap jellemzőt szemlélteti. 47

6.2. **Integrális kép.** Az integrális kép értéke az (x,y) pontban a tőle balra fent elhelyezkedő pixelek intenzitásértékeinek összege. 48

6.3. **Az integrális kép előnye a téglalap összeg számítása esetén.** 48

6.4. **Detektor kaszkád.** 53

6.5. **A "klikk", a "pillangó" és a "vénusz" spektrális képe.** 56

6.6. **A "klikk" szó felismerési eredményei.** Az x a kaszkádok, az y tengely a találatok száma. Felülről lefelé a 0, 1, illetve 2 szükséges szomszédos találat esete. 57

6.7. **A "pillangó" szó felismerési eredményei.** Az x a kaszkádok, az y tengely a találatok száma. Felülről lefelé a 0, 1, illetve 2 szükséges szomszédos találat esete. 58

6.8. **A "vénusz" szó felismerési eredményei.** Az x a kaszkádok, az y tengely a találatok száma. Felülről lefelé a 0, 1, illetve 2 szükséges szomszédos találat esete. 59

Irodalomjegyzék

[1] Weiye M. A., *Connectionist vector quantization in automatic speech recognition*, 1999, Ph.D. thesis, Katholieke Universiteit Leuven.

[2] Y. Amit, D. Geman, and K. Wilder, *Joint induction of shape features and tree classifiers*, (1997).

[3] Jarai Antal, *Számítógépes számelmélet*, 1998, Egyetemi jegyzet, http://compalg.inf.elte.hu/ ajarai/cnt.pdf.

[4] Szendrő Balázs, *Testre szerelt rf-környezeti intelligencia eszköz használat intelligens szövegbevitelre*, 2004, TDK dolgozat.

[5] L. E. Baum and J. A. Egon, *An inequality with applications to statistical estimation for probabilistic of markov procces and to a model for ecology*, Bull. Amer. Meterol. Soc **73** (1967), 360–363.

[6] L. E. Baum and G. R. Sell, *Growth functions for transformations on manifolds*, Pac. J. Math **27** (1968), 211–227.

[7] Sahar E. Bou-Ghazale and John H. L. Hansen, *Speech feature modeling for robust stressed speech recognition*, ICSLP vol.3 (1998), 887–890.

[8] M.J. R. Healy B.P. Bogert and J.W. Tukey, *The quefrency alanysis of time series for echoes: Cepstrum, psuedo-autocovariance, cross-cepstrum and saphe cracking*, Proceedings of the Symposium on Time Series Analysis (1963).

[9] A. P. Dempster, N. M. Laird, and D. B. Rubin, *Maximum likelihood from incomplete data via the em algorithm*, J. Roy. Stat. Soc. **39** (1977), 1–38.

[10] G. D. Forney, *The viterbi algorithm*, Proc. IEEE **61** (1973), 268–278.

[11] J. Frankel, *Linear dynamic models for speech recognition*, 2003, Ph.D. thesis, Center for Speech Technology Research, University of Edinburgh.

[12] William T. Freeman and Edward H. Adelson, *The design and use of steerable filters*, IEEE Transactions on Pattern Analysis and Machine Intelligence **13** (1991), no. 9, 891–906.

[13] Yoav Freund and Robert E. Schapire, *A decision-theoretic generalization of online learning and an application to boosting*, Computational Learning Theory: Eurocolt '95, Springer-Verlag, 1995, pp. 23–37.

[14] M.J.F. Gales and S.J. Young, *The theory of segmental hidden markov models*, Technical Report CUED/F-INFENG/TR. 133, Cambridge University Engineering Department, 1993, http://mi.eng.cam.ac.uk/reports.

[15] H. Greenspan, S. Belongie, R. Gooodman, P. Perona, S. Rakshit, and C. Anderson, *Overcomplete steerable pyramid filters and rotation invariance*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1994.

[16] H. Hermansky, *Perceptual linear prediction*, Journal of Acoustic Society of America 87(4) (1990), 1732–1752.

[17] W.J. Homes and M.J. Russel, *Probablisitic-trajectory segmental hmms*, Computer Speech and Language (1999), 3–37.

[18] Oscar Mayora Ibarra and Francesco Curatelli, *A brief introduction to speech analysis and recognition*, 2000, http://www.mor.itesm.mx/ omayora/Tutorial/tutorial.html.

[19] L. Itti, C. Koch, and E. Niebur, *A model of saliency-based visual attention for rapid scene analysis*, IEEE Patt. Anal. Mach. Intell. **20** (November 1998.), no. 11, 1254–1259.

[20] B. H. Juang, *Maximum likelihood estimation for mixture multivariate stochastical observations of markov chains*, 1985 **64** (AT&T Tech. J.), 1235–1249.

[21] B. H. Juang, S. E. Levinson, and M. M. Sondhi, *Maximum likelihood estimation for multivatiate mixture observations of markov chains*, IEEE Trans. Informat. Theory **32** (1986), 307–309.

[22] S. E. Levinson, L. R. Rabiner, and M. M. Sondhi, *An introducion to the application of the theory of probabilistic functions of a markov process to automaitc speech recognition*, Bell Syst. Tech. **62** (1983), 1035–1074.

[23] L. A. Liporace, *Maximum likelihood estimation fot the multivariate observations of markov sources*, IEEE Trans. Informat. Theory **28** (1982), 729–734.

[24] Karl Lippe, *Properties of analogue and digital signals*, 2002, http://www9.dw-world.de/rtc/infotheque/digital_signal/properties.pdf.

[25] Kiszlinger Melinda, *Szemdetektor*, 2004, Diplomamunka.

[26] Brian Van Osdol, *Cepstrum*, 2004, The Connexions Project.

[27] Edgar Osuna, Robert Freund, and Federico Girosi, *Training support vector machines: an application to face detection*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1997.

[28] C. Papageorgiou, M. Oren, and T. Poggio, *A general framework for object detection*, In International Conference on Computer Vision, 1998.

[29] J. Quinlan, *Induction of decision trees*, Machine Learning **1** (1986), 81–106.

[30] L. R. Rabiner, *A tutorial on hidden markov models and selected applications in speech recognition*, Proceeding of the IEEE **77** (1989), 257–286.

[31] L. R. Rabiner and B. H. Juang, *An introduction to hidden markov models*, IEEE ASSP Magazine (1986), 4–15.

[32] Anti-Veikko Ilmari Rosti, *Linear gaussian models for speech recognition*, 2004, Ph.D. thesis, Unversity of Cambridge.

[33] D. Roth, M. Yang, and N. Ahuja, *A snowbased face detector*, Neural Information Processing 12, 2000.

[34] H. Rowley, S. Baluja, and T. Kanade, *Neural network-based face detection*, IEEE Patt. Anal. Mach. Intell., vol. 20, 1998, pp. 22—38.

[35] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee, *Boosting the margin: a new explanation for the effectiveness of voting methods*, Ann. Stat. **26** (1998), no. 5, 1651–1686.

[36] H. Schneiderman and T. Kanade, *A statistical method for 3d object detection applied to faces and cars*, International Conference on Computer Vision, 2000.

[37] Mark D. Skowronsk and John G. Harris, *Improving the filter-bank of a classical speech feature extraction algorithm*, IEEE Intl Symposium on Circuits and Sysyems vol.IV (2003), 281–284.

[38] Mark D. Skowronski and John G. Harris, *Increased mfcc filter bandwith fot noise-robust phoneme recognition*, ICASSP (2002), 801–804.

[39] K. Sung and T. Poggion, *Example-based learning for view-based face detection*, IEEE Patt. Anal. Mach. Intell., vol. 20, 1998, pp. 39–51.

[40] K. Tieu and P. Viola, *Boosting image retrieval*, International Conference on Computer Vision, 2000.

[41] J.K. Tsotsos, S.M. Culhane, W.Y.K. Wai, Y.H. Lai, N. Davis, and F. Nuflo, *Modeling visual-attention via selective tuning*, Artificial Intelligence Journal **78** (October 1995), no. 1-2, 507–545.

[42] Paul Viola and Michael J. Jones, *Robust real-time object detection*, Technical Report CRL-2001-01, Cambridge Research Laboratory, 2001, http://www.hpl.hp.com/techreports/Compaq-DEC/CRL-2001-1.pdf, 2005.

[43] A. J. Viterbi, *Error bounds for convolutional codes and an asymptoticaly optimal decoding algorithm*, IEEE Trans. Informat. Thory **13** (1967), 269–269.