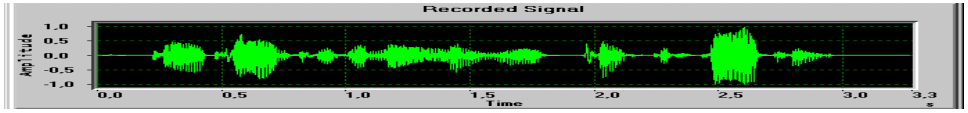
Beszédfelismerés

Bálint Enikő

Tartalom

1. **Bevezetés**



2. Történelmi áttekintés
3. Alapfogalmak a beszédfelismerésben
4. Hogyan működik?
5. BF-ek osztályozása
6. BF módszerek
7. Példa Java BF-applikációra
8. Linkek

1. Bevezetés

1.1 Definíciók

1.2 Alkalmazási területek

1.3 A BF nehézségei

1.1 Definíciók

- **Beszédfelismerés (BF):** az a folyamat, melynek során a beszédfelismerő gép azonosítja a kiejtett beszédjeleket és átalakítja ezeket szöveggé, vagy más, számítógép által feldolgozható adattá.
- (Automatic) Speech Recognition – ASR
- Miért van szükségünk rá?
- Hatékony kommunikáció a számítógéppel

- **Beszéd**: *hangok* kibocsájtása (akusztikus hullámok) = *fonémák*
- Nem csupán fonémák sorozata, hanem fontos a kontextus is: hangsúlyozás, hanglejtés stb.
- A BF szintjén *beszédjlek* (speech signal) = fonémák elektronikus képe

1.2 Alkalmazási területek

- Diktálás: a leggyakrabban használt terület napjainkban
- Hallássérültek oktatása
- Mobiltelefonok (hangtárcsázás stb.)
- Tartalomkeresés audiofájlokban

- Otthoni gépesítés (domotics): fény és légkondicionáló vezérlése,
- ajtó, ablakok bezárása,
- riasztóberendezés bekapcsolása,
- multimédiás eszközök elindítása,
- automatikus virágöntözés és háziállat etetése ☺

Diktálás:

- 1. hangadat bevitele
- 2. szöveggé alakítás
- 3. a szöveg javítása, nyelvtani, helyesírási korrekció (egyelőre még emberi beavatkozás szükséges)
- a diktálás egy alfaja: orvosi dokumentációk, leletkészítés
- Magyarországon már sikeresen használják

1.3 A beszédfelismerés nehézségei

- A BF rendszerek sok problémával szembesülnek → befolyásolják a program teljesítményét
- Ok: az emberi nyelv, a kommunikáció komplexitása
- Ma már a fejlettebb BF-ek képesek ezeknek a problémáknak a kezelésére
- Az embernek van bizonyos háttértudása a személyről, akivel beszél, a témáról, a másodlagos jelentésekről, a testbeszédről stb. ←→ számítógép

- A beszélők közötti különbség: mindenki másképp ejt ki egy hangot
- Egy beszélő sem képes kétszer pontosan ugyanazt a hangot produkálni
- Különböző beszédstílusok, hangmagasság, beszéd gyorsasága
- Anatómiai felépítés: beszédhibás emberek stb.
- **Megoldás**: lényegkiemelés
- Lehetséges eljárás: lineáris predikció, Fourier transzformáció stb.

- Zajos környezet

Megoldás: léteznek zajkiszűrő algoritmusok (pl. Kálmán szűrő)

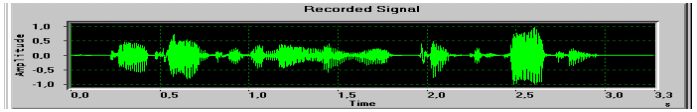
  - Több mikrofon használata

Még megoldatlan problémák:

- Beszélt nyelv ←→ írott nyelv
- Korlátlan adatmennyiség

Tartalom

1. Bevezetés
2. **Történelmi áttekintés**
3. Alapfogalmak a beszédfelismerésben
4. Hogyan működik?
5. BF-ek osztályozása
6. BF módszerek
7. Példa Java BF-applikációra
8. Linkek

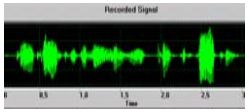


2. Történelmi áttekintés

- BF: eredetileg mozgássérültek számára fejlesztették ki
- **1936:** a AT&T's Bell Labs cég megalkotja az első elektronikus beszédszintetizáló gépet, a Vodert
- **1950-es évek**: a legkorábbi kísérletek automatikus beszédfelismerés terén
- **1969:** John Pierce azt állította, hogy az automatikus beszédfelismerés nem valósítható meg még néhány évtizedig, mert ehhez szükség van mesterséges intelligencára.

- **1970-es évek eleje:** Lenny Baum feltalálja a Rejtett Markov Modell (Hidden Markov Model (HMM)) alkalmazhatóságát BF-ek esetén.
- A HMM elvet hamarosan minden vezető beszédfelismeréssel foglalkozó cég alkalmazza
- **1985:** a Kurzweil cég, első 1000 szavas beszédfelismerő, a Kurzweil Voice System
- **90-es évek vége:** sok BF, jobb tulajdonságok, hozzáférhető ár
- (**1997:** a Microsoft 45 millió dollárt fektetett egy olyan projektbe, ami lehetővé tette a Microsoft és Corel Office beszédvezérlését)
- **Napjainkban** széles körben alkalmazzák

Tartalom

1. Bevezetés
2. Történelmi áttekintés
3. **Alapfogalmak a beszédfelismerésben** 
4. Hogyan működik?
5. BF-ek osztályozása
6. BF módszerek
7. Példa Java BF-applikációra
8. Linkek

3. Alapfogalmak a beszédfelismerésben

Kijelentések (Utterances)

- Két szünet közötti folyamatos beszéd
- Egy egységes jelentésű elemnek tekinti a BF
- Állhat egy szóból, vagy akár több szóból is.

Beszélőfüggőség (Speaker Dependence)

- A BF egy meghatározott beszélőre van tervezve.
- Pontosak az illető beszélő esetében, de kevésbé pontosak más beszélők esetén.
- Általában lehetőség van a rendszer tanítására.

Szótár (Vocabularies, Dictionaries)

- Szavak és kijelentések listája, amit a BF rendszer képes azonosítani.
- Egy szónak több jelentése is lehet

Pontosság (Accuract)

- Egy BF képessége = pontosságával
- Milyen jól ismeri fel a kijelentéseket.

Tanítás (Training)

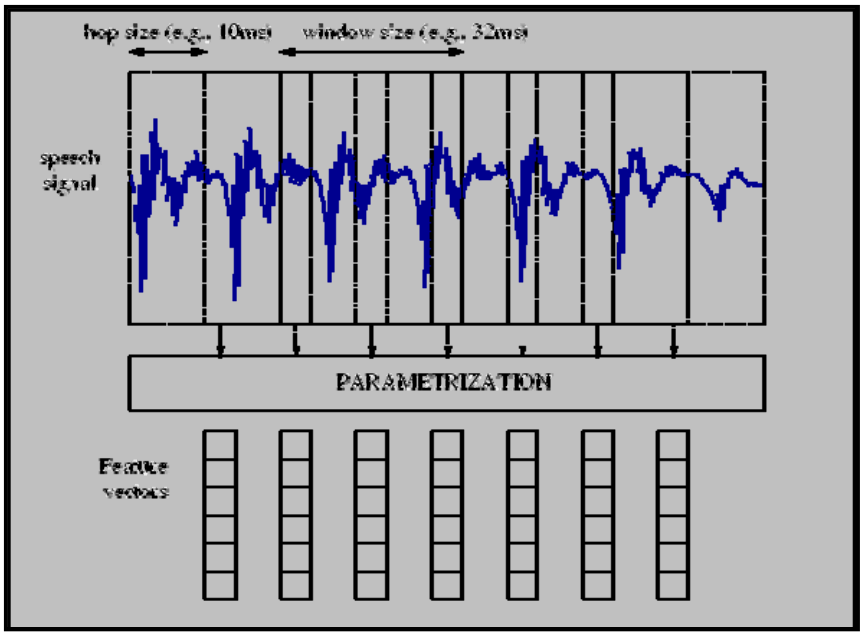
- Alkalmazkodik a felhasználóhoz.
- A BF tanítása növeli annak a pontosságát

Kiejtés (Pronunciations)

- Egy szónak többféle kiejtése is lehet (pl. az angolban a ‘the’)

Nyelvtan (Grammars)

- Megadja a szavak használatának, sorrendjének a helyességét
- Egy speciális szintaxisból, szabályok sorozatából áll
- Nem minden BF használja a nyelvtanokat

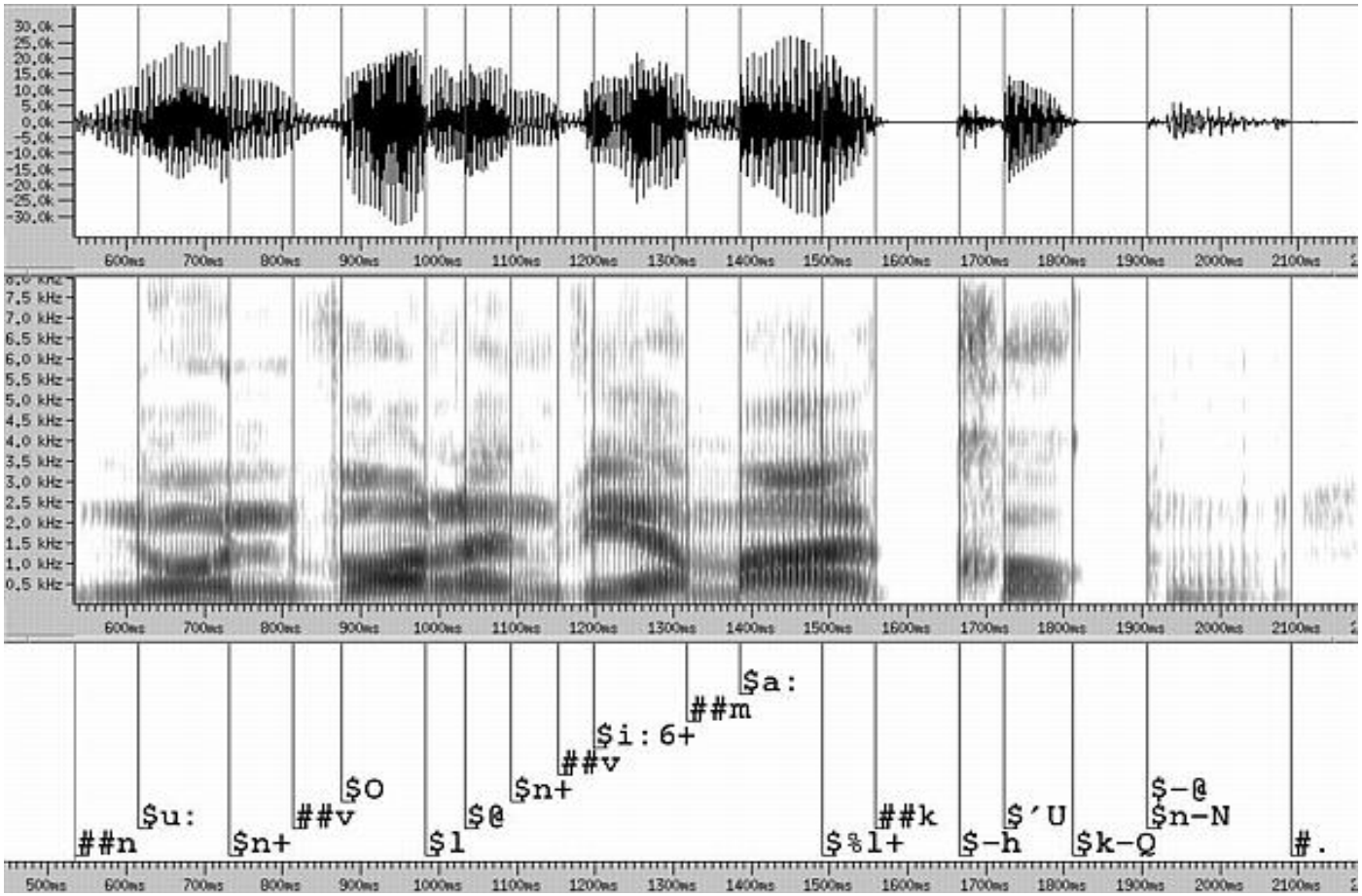


Jellegvektor vagy lényegvektor (Feature vector): egy adott n dimenziós vektor, amely olyan jellemzőket tárol, amelyek alkalmasak felismerésre, szegmentálásra.

- Az ablakokra (időszegmensekre) osztott beszédjel jellemzőit tárolja

Spektrogram:

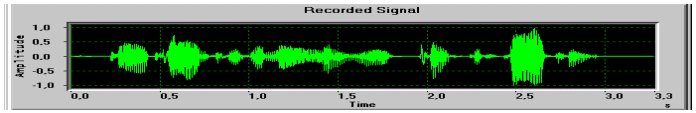
- A beszédjel(ek) frekvenciaváltozásának időbeli vizuális megjelenítése
- Két- vagy háromdimenziósan ábrázolják: idő, frekvencia (+ energia)
- Gyakran használnak különböző színeket az energia kimutatására
- Fourier transzformációval hozható létre



- Példa kétdimenziós spektogramra (középső rész)

Tartalom

1. Bevezetés
2. Történelmi áttekintés
3. Alapfogalmak a beszédfelismerésben
4. **Hogyan működik?**
5. BF-ek osztályozása
6. BF módszerek
7. Példa Java BF-applikációra
8. Linkek



4. Hogyan működik?

- A BF **általában** a következő fázisokra bontható le:

1.) Hangfelvétel. A beszédjel továbbítása a BF-nek

2.) Előzetes szűrés vagy lényegkiemelés (Pre-Filtering):

- A beszédjelbõl megkísérli meghatározni a beszéd tartalmát hordozó mennyiségeket, és kiküszöbölni a felismerés szempontjából érdektelen információkat (zaj, fázis, torzítások).
- Törekvés az invariáns elemek kiemelésére
- Kimenet: egy adott dimenziójú lényegvektor-sorozat

3.) Keretezés (Framing and Windowing):

- Ablak mérete ms-ban (10-30 ms), 50%-os fedésben
- A lényegvektor-sorozat feldarabolása

4.) További szűrés:

- Nem minden esetben jelenik meg, van, amikor az előzetes szűrés elégséges.
- Sokszor itt megy végbe az időillesztés és a normalizálás.

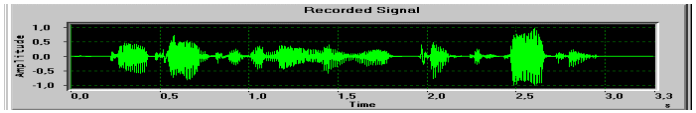
5.) Összehasonlítás és illesztés (osztályozás):

- Felismerni a kijelentést.
- Ez legtöbbször valamilyen mintával való összehasonlítást jelent (például két jellegvektor összehasonlítása)
- Minden alkalmazott módszer egy valószínűségi és pontossági illesztést végez.

6.) Kimenet (pl. írott szöveg), feldolgozás

Tartalom

1. Bevezetés
2. Történelmi áttekintés
3. Alapfogalmak a beszédfelismerésben
4. Hogyan működik?
5. **BF-ek osztályozása**
6. BF módszerek
7. Példa Java BF-applikációra
8. Linkek



5. BF-ek osztályozása

- Többféle osztályozási szempont is létezik

1. A szótár nagysága szerint
2. A megvalósítás módja szerint
3. A feldolgozott kijelentések típusa szerint

5.1 A szótár mérete szerint

a. **kis (kötött, zárt) szótáras**, kb. 100 szó
b. **nagy szótáras** (kötetlen, nyílt szótáras), 20-80000 szó

- Lehet finomítani pl. közepes méretű

5.2 A megvalósítás módja szerint

- hogyan kezeli a bemeneti hangadatot, illetve, hogy milyen algoritmusokat használ

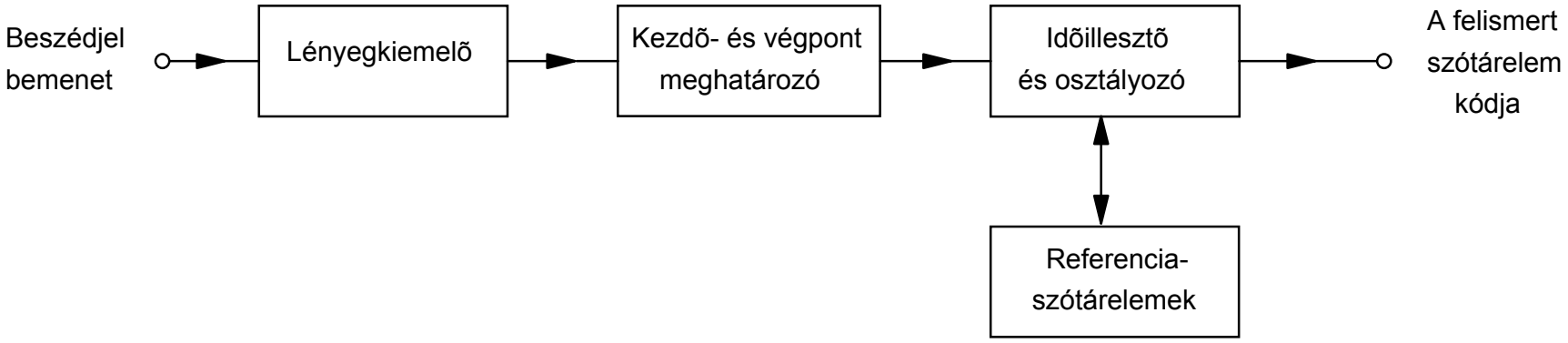
a. **Mintafelismerő**
   - Napjainkban főleg ezt használják
   - Ilyen pl. Rejtett Markov Modell

b. **Akusztikus-fonetikus**
   - Pl. Neuronális hálós és a ismeretalapú (tudásalapú) módszer

5.3 A feldolgozott kijelentések típusa szerint

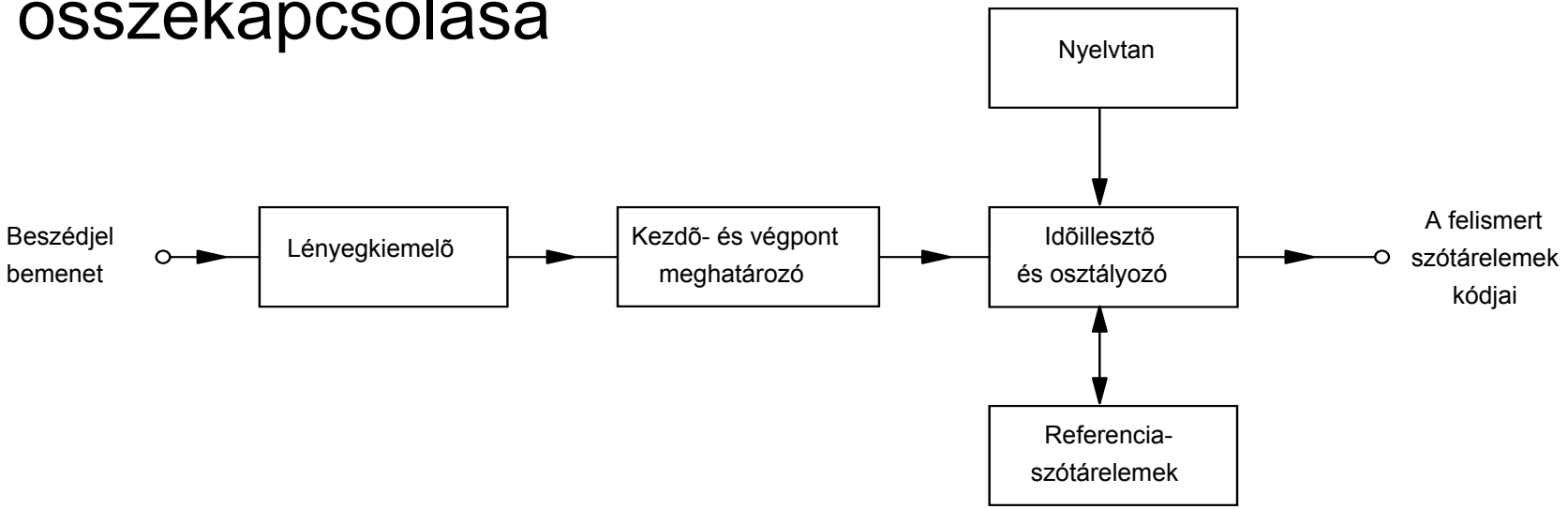
a.) Izolált szavak, kijelentések felismerése

- egyszerre csak egy kijelentést vár → a beszélő kell várakozzon
- Egy szótárelemet határoz meg a szótárból



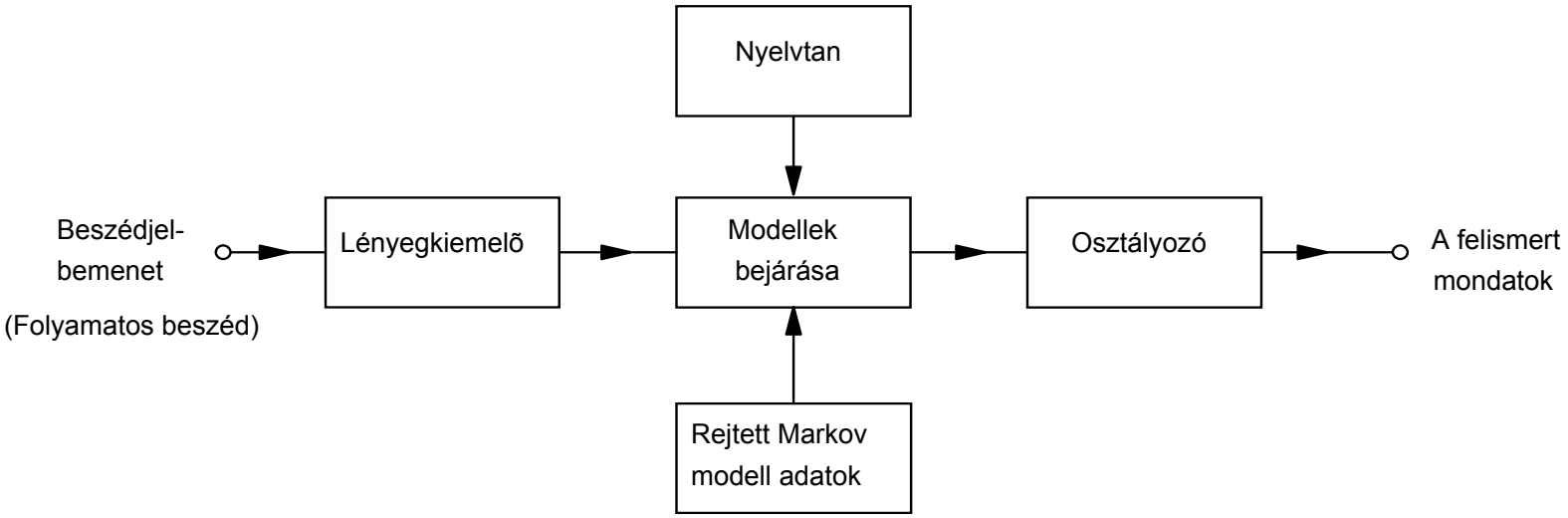
b.) Összekapcsolt kijelentések

- Hasonlít az előzőhöz, de a hallgatás ideje minimális
- A BF nem ismeri a szavak határát → minden kombinációt figyelmbe kell vegyen
- Megjelenik a nyelvtan is: a szavak összekapcsolása



c.) Folyamatos beszéd

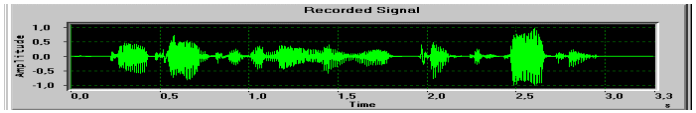
- Nehéz ilyen rendszereket alkotni
- Valószínűségszámítási algoritmusok pl. HMM



d.) Hangfelismerés

- Különböző felhasználók hangjainak a megkülönböztetése

Tartalom

1. Bevezetés
2. Történelmi áttekintés
3. Alapfogalmak a beszédfelismerésben
4. Hogyan működik?
5. BF-ek osztályozása
6. **BF módszerek** 

7. Példa Java BF-applikációra
8. Linkek

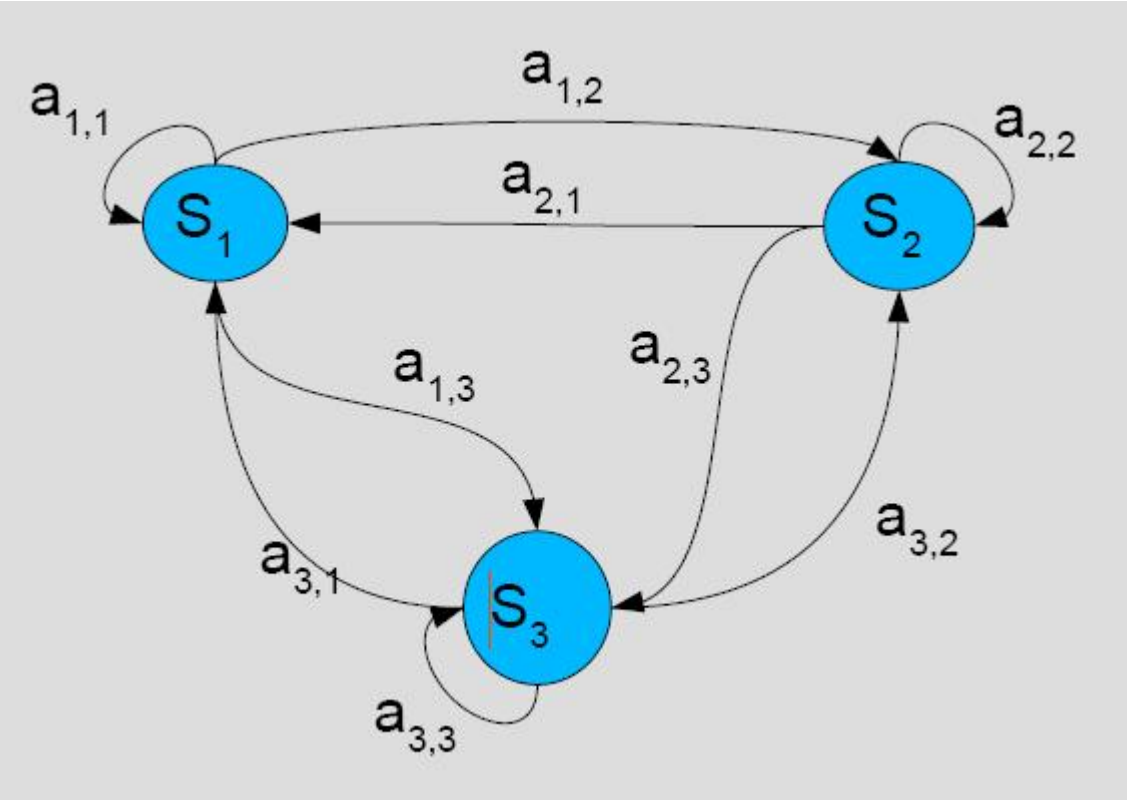
6. BF módszerek

Rejtett Markov Modell (HMM)

6.1 Markov Modell
6.2 Markov Modell példa
6.3 Rejtett Markov Modell
6.4 HMM példa
6.5 HMM összegzés

6.1 Markov Modell

- Állapotok, állapotátmenetek



Forrás:digitus.itk.ppke.hu/~flugi/hmm.pdf

6.2 Markov Modell példa: időjárás

- Állapotok: S={S1:Napos, S2:Borult, S3:Esős}
- Állapotátmenetek mátrixa ($S_i \rightarrow S_j$):

$$A=\{a_{i,j}\}=\begin{vmatrix} 0,4 & 0,3 & 0,3 \\ 0,2 & 0,6 & 0,2 \\ 0,1 & 0,1 & 0,8 \end{vmatrix}$$

- Valószínűségeket tárol
- Kérdés: adott állapot-sorozat valószínűsége
- Pl: Mi az esélye, hogy egy Napos állapot után {Napos, Esős} sorozat következzen?
- P({S1,S1,S3}|S1)=1*a1,1*a1,3=1*0.8*0.3=0.24

6.3 Rejtett Markov Modell (HMM)

- Fonéma, félszótag vagy szó = **állapotok** sorozata
- A beszéd egy **folyamat**, ez a folyamat minden t= 1, 2, ... időpillanatban egy $S = \{S_1, S_2, \ldots, S_N\}$ állapotban van (q_t)
- **A** mátrix – állapotátmenetek valószínűsége
- A modell az adott állapotban **megfigyeléseket** tesz, ($O=(O_1,\ldots O_M)$, M - megfigyelések száma)
- **B** mátrix – a megfigyelések valószínűsége

- **Rejtett:** a megfigyelés az állapotok valamilyen sztochasztikus függvénye, így a belső állapotok rejtve maradnak.
- Az előbbi példában a megfigyelések konkrét, fizikai állapotokra vonatkoztak, a rejtett MM-ben az állapotokhoz rendelt megfigyelések valószínűséget jelölnek
- A folyamat minden lépésben új állapotba kerül a valószínűségi értékek alapján.
- Az állapotátmenetek a BF-ben ismertek, nagy adatbázisokból kinyerhetők
- Általános esetben $q_t = S_i$ valószínűsége függhet attól, hogy a t-t megelőző pillanatokban milyen állapotban volt a rendszer.

6.4 HMM példa

Éremfeldobás

- Úgy, hogy nem láthatod, valaki érmeket dobál fel
- Nem mondja meg, hogy éppen mit csinál, csak az eredményt : fej vagy írás
- → *rejtett* éremdobálás, megfigyelés: a fej vagy írás eredmények sorozata
- Pl. O = F F I F I I ... (F – fej, I-írás)
- Megoldandó probléma: hogyan építünk fel egy HMM-et, ami megmagyarázza a kapott sorozatunkat

- Az első lépés annak az eldöntése, hogy mit tartunk állapotnak
- Utána el kell dönteni, hogy hány állapot lesz a modellben
- Ha csak egyetlen érmét veszünk, akkor egy kétállapotos modellt tudunk felrajzolni, amelyikben egy állapot az érmének az egyik felét jelenti → a MM megfigyelhető

- Ha két érmét veszünk és két állapotot, amiben az egyik állapot az egyik érmére, a másik állapot a másik érmére vonatkozik
- Az érméket egyszerre hajítjuk fel
- Mindegyik állapotot egy valószínűségi eloszlás jellemzi, az állapotátmeneteket le lehet írni egy megfelelő mátrixszal

A HMM három alapfeladata:

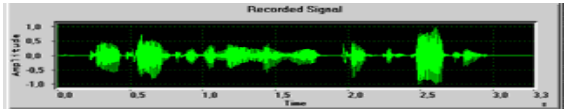
1. Adott O megfigyeléssorozat és β modell alapján hogyan számítjuk ki hatékonyan a P(O | β) valószínűséget
2. Adott O megfigyeléssorozat és β modell alapján hogyan számítunk ki egy megfelelő q állapotsorozatot, ami optimális az adott esetben (a modell rejtett részének a meghatározása, fontos folyamatos beszéd felismerése esetén)
3. Hogyan állítsuk be adott β modell esetén a paramétereket (A, B), hogy maximalizáljuk a P(O | β) valószínűséget (a modell „tanítása”)

6.5 HMM - összegzés

- Ahhoz, hogy egy HMM sikeresen működjön, beszédfelismerési vagy egyéb rendszerekben, az A és B mátrixok pontos becslése szükséges.
- Ezeket az értékeket adatbázisok tartalmából ki lehet számítani megközelítőleg
- A megoldás a statisztika, a maximum likelihood módszer.

Tartalom

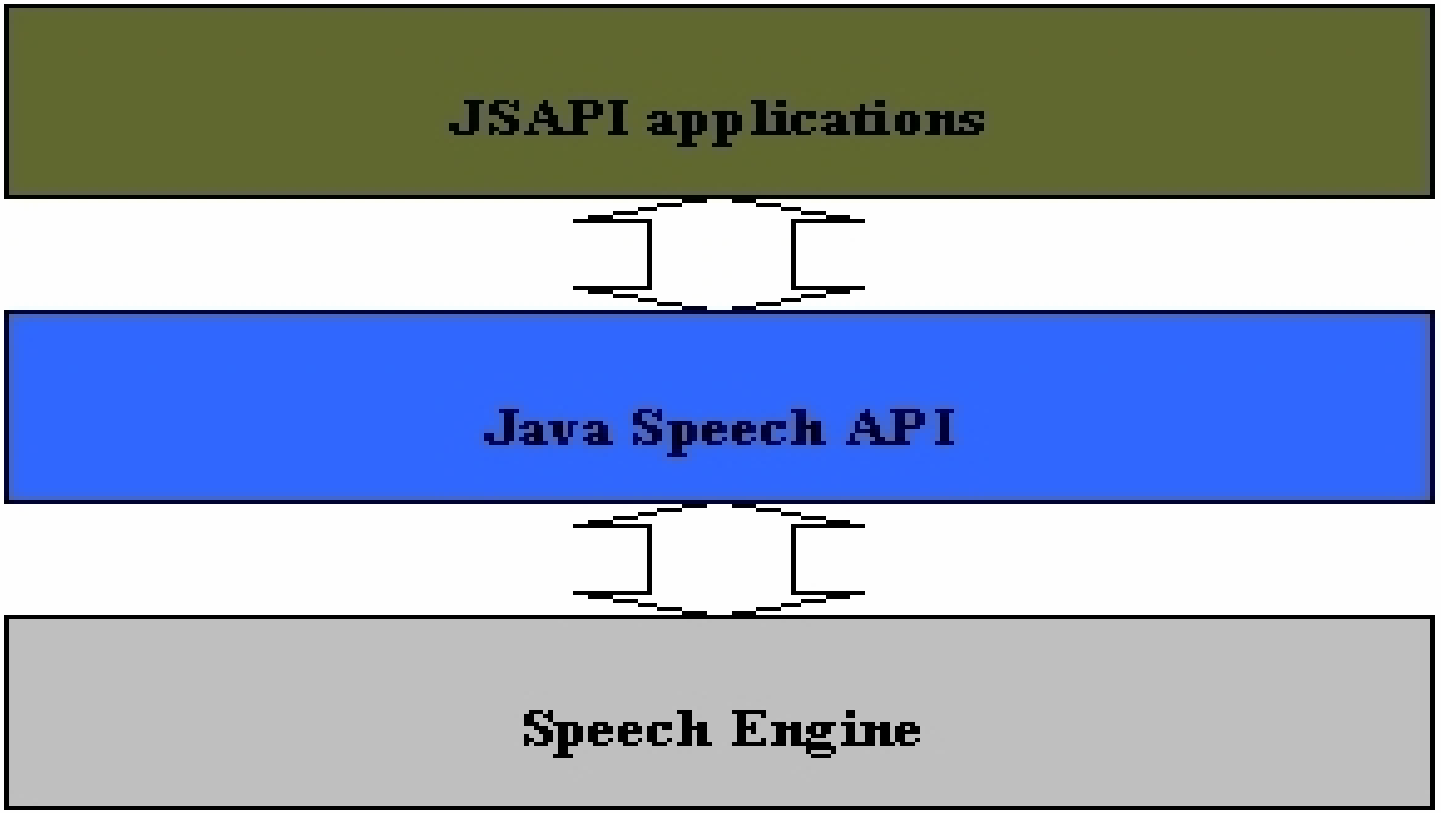
1. Bevezetés
2. Történelmi áttekintés
3. Alapfogalmak a beszédfelismerésben
4. Hogyan működik?
5. BF-ek osztályozása
6. BF módszerek
7. **Példa Java BF-applikációra**
8. Linkek



Java BF

- Speech engine (pl. Microsoft SAPI4 v. SAPI5)
- Java Speech API (JSAPI) → jsapi.jar
  - import javax.speech.*;
  - import javax.speech.recognition.*; (van synthesis is)
- Egy JSAPI-t megvalósító alkalmazás (sok ingyenes is van pl. Sphinx-4)

Java Speech API



Egy egyszerű BF projekt

- A következőkben egy egyszerű Java BF létrehozását mutatom be, majd rátérek a saját alkalmazásomra
- Az én programom két részből áll:

1. Nyelvtan definiálása: txt fájl
   - Meghatározza, hogy mit ismer fel a BF
2. A java kód, ami ezt a nyelvtant felhasználja

A JSpeech Grammar formátuma (JSGF)

//header

- #JSGF V1.0 ISO8859-1 en;
- grammar gram1;

//body

- public <valami> = Mary loves John | apple;

A java BF létrehozásának lépései

- A BF deklarálása:

  Recognizer rec;
- A BF létrehozása a Central osztály egy metódusával:

  rec = Central.createRecognizer(new EngineModeDesc(Locale.ENGLISH));
- Angolnyelvű BF

- A BF elindítása, ellenőrzi, hogy a BF-nek minden forrása megvan-e:

  rec.allocate();

- A nyelvtan fájlból való beolvasása és engedélyezése:

  FileReader reader = new
  FileReader(“grammar.txt”);

  RuleGrammar gram = rec.loadJSGF(reader);

  gram.setEnabled(true);

- A BF-hez hozzá kell rendelni egy listenert (megoldás később):

  rec.addResultListener(new MyResultListener());
- Ha a nyelvtanban bármilyen változás volt, ezt jelezni kell:

  rec.commitChanges();
- A következő két metódus nélkül a BF nem indul el (nem aktív):

  rec.requestFocus();

  rec.resume();

- A listener osztály:

  Class MyResultListener extends ResultAdapter
- Egyetlen metódust implementál: azt az esetet, amikor a felismerés sikeres volt
- RESULT_ACCEPTED eseményt kap

  public void resultAccepted(ResultEvent e) {}

- Az eredményből (*Result r*) kivesszük a legpontosabb találatokat és egy vektorban tároljuk:

  ResultToken tokens[] = r.getBestTokens();
- Egy elemre való hivatkozás: *tokens[i].getSpokenText()*
- A BF megszüntetése és kilépés

  rec.deallocate();

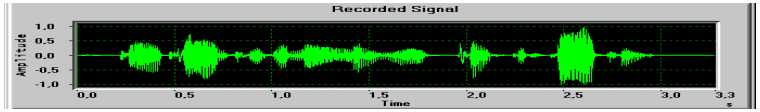
  System.exit(0);

Saját BF

- A projekt egy beszédvezérléses e-mailküldő rendszert szimulál (e-mailküldő funkció nélkül)
- A program képes különböző utasításokat végrehajtani beszéd segítségével:
  - Egy szöveges mezőbe diktálás
  - Igen/nem válaszok eldöntése
  - Kilépés a programból
- Csak beszéd útján működik

Tartalom

1. Bevezetés
2. Történelmi áttekintés
3. Alapfogalmak a beszédfelismerésben
4. Hogyan működik?
5. BF-ek osztályozása
6. BF módszerek
7. Példa Java BF-applikációra
8. **Linkek**



Ingyenes szoftverek + tutorial

- **XVoice**
- http://www.zachary.com/creemer/xvoice.html
- **Sphinx-4**
- http://cmusphinx.sourceforge.net/sphinx4/
- **GVoice**
- Gtk/GNOME applikációk vezérlésére
- **FreeTTS**
- http://freetts.sourceforge.net/docs/index.php

- **JavaTM Speech API Programmer's Guide**
- http://java.sun.com/products/java-media/speech/forDevelopers/jsapi-guide/index.html

Néhány hasznos cím az interneten

- **BF-ről általában:**
  - http://tldp.org/HOWTO/Speech-Recognition-HOWTO/introduction.html (angol)
  - http://www.stillhq.com/diary/asr-short.pdf (angol)
- **HMM:**
  - http://web.mit.edu/6.435/www/Rabiner89.pdf (angol)
  - http://cslu.cse.ogi.edu/HLTsurvey/ch1node7.html (angol)
  - http://www.cs.brown.edu/research/ai/dynamics/tutorial/Documents/HiddenMarkovModels.html (angol)
  - http://digitus.itk.ppke.hu/~flugi/hmm.pdf (magyar)

Köszönöm a figyelmet.