Zeneszám felismerés

Mérési segédlet

BME

2016. január

Tartalomjegyzék

1 Bevezetés

Napjainkban a multimédia tartalmak létrehozása és felhasználása igen széleskörű. Változatos formában merülnek fel olyan problémák, amelyekben audio anyagokat kell összehasonlítani egymással. Ilyen konkrét problémák például:

- Zeneszám azonosítása részlet alapján, a felhasználó telefonjának mikrofonjával rögzítve a környezetből, pl. Shazam alkalmazás
- Zeneszám azonosítása műsorfolyamban, rádióból, tévéműsorból, internetes tartalomból, hatóság részére, szerzői jogi szempontból
- Szignálok, reklámok azonosítása műsorfolyamban, szerződés szerinti reklám gyakoriság ellenőrzése és egyéb statisztikák céljára
- Audio adatbázisban duplikált elemek keresése, a redundáns tárhelyhasználat csökkentése és a rajtuk futó algoritmusok helyes működése érdekében
- A felhasználó saját bemondásaival betanított egyszerű beszédfelismerés

A felsorolt problémák megoldására használt módszereknek különböző mértékű torzítás- és zajtűrési követelményeket kell kielégíteniük. A mikrofonnal felvett zeneszám azonosításánál nagy energiájú környezeti zajokra, a mikrofon, a hangszóró, és a köztük lévő tárgyak által okozott torzításra kell számítani. A csak elektronikus rendszeren keresztül haladó jelek esetében különböző analóg csatornatorzításokra, audió kodekek torzítására, és megakadások, keret kiesések fordulhatnak elő. A saját szavakkal tanított beszédfelismerésnél ezeken kívül még időbeli torzulás is várható, mivel nem mindig feltétlenül ugyanolyan tempóban mondja be az ember ugyanazt a szót.

Fontos követelmény még a lehetőleg minél gyorsabb működés. Ezt például úgy lehet elérni, hogy valamilyen módon kivonatot készítünk a hanganyagokból, ami az eredeti hanganyagnál lényegesen kisebb méretű, de az azonosításhoz szükséges jellemzőkkel rendelkezik.

A mérési feladatban olyan módszereket fogunk megvizsgálni, amik főleg az első 4 problémára használhatók, mivel az összehasonlítandó hanganyagok közötti eltérésben a zajt, torzítást eltűrik, de az időbeli torzulást, sebességbeli különbséget nem.

Az egyik mérésben szereplő módszer az MFCC lényegkiemelés. Ennek egyik lényeges eleme a frekvenciatartományba való áttérés, a diszkrét Fourier-transzformáció. Meg fogjuk vizsgálni pusztán a DFT-nek a hangminta azonosításra való alkalmazhatóságát is.

A másik módszer, amit meg fogunk vizsgálni, az ujjlenyomat számítás és illesztés módszere. Ez szintén DFT-vel indul, de a továbbiakban teljesen eltérő elv szerint működik, a spektrogramban található csúcsok egymáshoz képesti helyzetével képez úgynevezett landmarkokat, amik összessége adja az audio ujjlenyomatot.

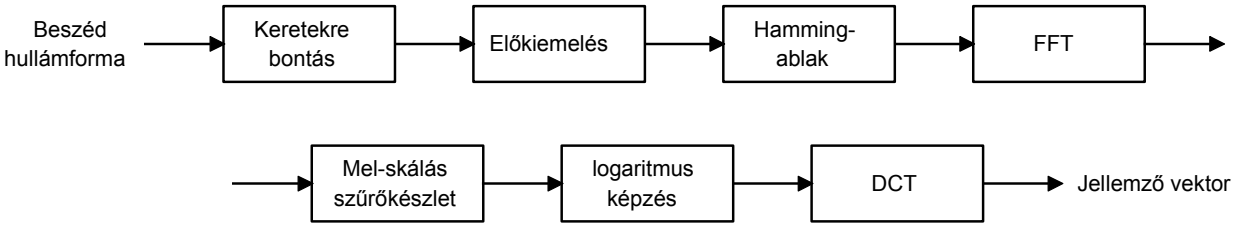
A 2. és 3. részben az MFCC és az ujjlenyomatos módszer elvének ismertetése következik.

2 Az MFCC lényegkiemelés

A gépi hangfeldolgozásban gyakori követelmény, hogy a hanganyagot olyan transzformációnak vessük alá, ami megőrzi azt a változatosságot, amit az emberi fül is érzékel, és az emberi agy fontos információnak tart, de eldobja az adott hanganyagnak az emberi fül és agy

számára érdektelen tulajdonságait, jellemzően például a zajokat. Praktikus követelmény még, hogy a létrehozott, lényegtelen információtartalommal csökkentett adathalmaz méretét tekintve is számottevően csökkenjen. Az ilyen hang-átalakításokat lényegkiemelésnek nevezzük.

Az egyik legszéleskörűbben használt lényegkiemelési eljárás a Mel-Frequency Cepstral Coefficients (röviden MFCC). Ennek blokkvázlata a 2.1. ábrán látható.



2.1. Ábra: MFCC lényegkiemelés blokkvázlata

A feldolgozási lánc bemenetére a feldolgozni kívánt hanganyag (beszéd vagy más, emberi fogyasztásra szánt hang, pl. zene) hullámformájának mintavételezett mintái érkeznek sorban. A teljes lánc kimenetén az úgynevezett jellemző vektorok jelennek meg bizonyos időközönként. A vektorok dimenziószáma szokásosan 13, a (bejövő hullámformán mért) időköz, ami szerint követik egymást, tipikusan 10 ms. De ezektől az értékektől el is lehet térni.

A MFCC lényegkiemelés lépései:

2.1 Keretekre bontás

A bejövő jel általában 16 kHz-es mintavételezésű, 16 bites, ha telefonhálózatról érkezik a jel, akkor elég a 8 kHz mintavételi frekvencia is. A jelet 25 ms-os keretekre bontjuk fel. Ez azért szükséges, mert a hangjel folyamatosan változik, és ezen változásokat nyomon szeretnénk követni és a feldolgozás blokkokban történik. Túl nagy keretméret esetén a gyors valtozások kiátlagolódnak, túl kis keretméret esetén pedig a hangjel jellemzőit pontatlanul tudjuk csak meghatározni.

A keretek 10 ms-onként követik egymást, tehát átlapoltak, azaz a következő keret kezdete az aktuális keret vége elé esik. Erre azért van szükség, hogy a feldolgozás során a hang jellemzőinek változását jól nyomon tudjuk követni.

2.2 Előkiemelés

Az előkiemelés az alacsony frekvencájú komponenseket elnyomja a jelben, míg a magasakat kiemeli. A használt előkiemelő egy elsőfokú FIR szűrő, amelynek átviteli függvénye:

$$W(z) = 1 - 0.95z^{-1}$$

2.3 Ablakozás

A diszkrét Fourier transzformáció (DFT) előtt ablakolni kell, mert nem periódikus jellel dolgozunk. Nem mindegy azonban, hogy az ablak milyen formájú. A legegyszerűbb ablak, a négyszögablak, azért nem jó, mert pl. egy tiszta szinusz jel esetén négyszög ablakot használva a DFT-vel előállított spektrum "szétkent" lesz, nem csak a szinuszjel frekvenciájának megfelelõ helyen találunk vonalat, azaz nullától különbözõ értéket. A Hamming ablak azonban a spektrumot "élesíti", mert szűri a spektrumot. Az ablakfüggvénnyel való szorzás az időtartományban konvolúciónak felel meg a frekvenciatartományban, a véges konvolúció pedig azonos egy FIR

szűrővel való szűréssel. Tehát az ablakolás a jel Fourier transzormáltját szűri.

A Hamming ablak függvénye:

$$h[n] = 0.54 - 0.46 \cos \frac{2\pi n}{N}$$

ahol n = 0 ... N-1, és N az ablak mérete.

2.4 Diszkrét Fourier transzformáció

A diszkrét Fourier transzformációval térünk át időtartományból frekvenciatartományba. Ez azért szükséges, mert a beszédhangok vagy zenei dallam jellemzői csak a jel spektrumán ismerhetők fel. Ezen kívül a bemenő jel számtalan torzulást szenved: véletlenszerű fázistolás a különböző frekvenciákon, additív zaj, torzítás (konvolúciós zaj), és ezeket frekvenciatartományban, illetve további transzformációk segítségével lehet csak kiküszöbölni.

A DFT-t gyors algorítmussal (FFT) érdemes számolni, ez ugyanis összehasonlíthatatlanul gyorsabb. A keletkezett komplex spektrum abszolút érték négyzetét tartjuk csak meg, a fázisinformáció a hangjel információtartalma szempontjából lényegtelen, sõt csak zavaró tényező.

2.5 Mel-skála szerinti sávszűrőkészlet

Az ember hangérzékelése a frekvencia függvényében változik. Magasabb frekvenciákon szélesebb frekvenciaközű hangokat képes csak megkülönböztetni, mint alacsonyabb frekvenciákon. Ez a megkülönböztetõ képesség (frekvencia felbontás) 1000 Hz alatt közelítőleg lineárisan változik, míg efölött logaritmikusan (tehát 1000 Hz fölött a sávok szélessége a frekvencia függvényében exponencilisan növekszik). Ezt hívják mel-skálának. Mivel emberi fül számára készített hangokkal dolgozunk, ezért célszerű a frekvenciákat is az emberi hallásnak megfelelő módon kezelni. Ez a skála tulajdonképpen azt mutatja, hogy a tartalomra jellemző információ a felismerés szempontjából milyen sűrűséggel helyezkedik el a frekvenciatengely mentén.

A mel-skála képlete:

$$f_{mel} = 2595 \cdot \log_{10}\left(1 + \frac{f_{lin}}{700\,Hz}\right)$$

A MFCC számításnál a DFT-vel nyert energiaspektrum értékeket egy frekvenciában mel-skála szerint egyenletesen elosztott sávszűrőkészlettel a sávszűrőknek megfelelő számú (pl. 40) számmá alakítjuk. Egy sávszűrőt egy háromszög-ablakként kell elképzelni, a szűrő alkalmazása pedig a teljesítményspektrum szorzása az ablakkal, majd az értékek összegzése.

2.6 Logaritmusképzés és diszkrét koszinusz transzformáció

A jelfeldolgozás utolsó két lépése a kepsztrum meghatározására szolgál. A "hagyományos" kepsztrumot lineáris skálázású logaritmikus spektrumból (abszolútérték DFT logaritmusából) inverz DFT-vel számítják. (A „cepstrum” elnevezés egy mesterségesen alkotott szó, egy szójáték. A „spectrum” szóból alkották annak érzékeltetésére, hogy a spektrumból képződik inverz DFT-vel.) Ezzel szemben az úgynevezett mel-kepsztrumot a fent leírt mel-skálájú szűrőkészlet logaritmikus kimenetéből számítják DFT-vel vagy diszkrét koszinusz transzformációval (DCT). Ez utóbbi transzformáció a képfeldolgozásban használatos, és fontos tulajdonsága, hogy a bemeneti jel fázisát megőrzi, és szemben a DFT-vel, csak valós értékeket

szolgáltat. (A bemeneti jel itt nem időtartománybeli, hanem logaritmikus spektrumtartománybeli. Míg a hangjel időfüggvénye szinusz-komponeneseinek a fázisa érdektelen számunkra, addig a logaritmikus spektrum szinusz-komponeneseinek a fázisa fontos információt hordoz például a beszédformánsok helyzetére, illetve a zene dallamára vonatkozólag.)

A DCT együtthatóinak meghatározása:

$$c_m = \sum_{i=0}^{M-1} f_i \cos\left(\frac{m(i-0.5)\pi}{M}\right)$$

ahol M a sávszűrők száma. Nem számítjuk ki az összes DCT együtthatót, csak 13-at. Ezzel megkaptuk a hangminták összehasonlításához szükséges jellemző vektort. (A DCT alkalmazásának valódi célja, hogy dekorrelálja a bemenetül kapott vektort. Így a DCT transzformált vektor a magasabb dimenziók elhagyásával is hatékonyan reprezentálhatja az eredeti vektor lényegi információtartalmát).

2.7 Hangminta összehasonlítás MFCC együtthatókkal

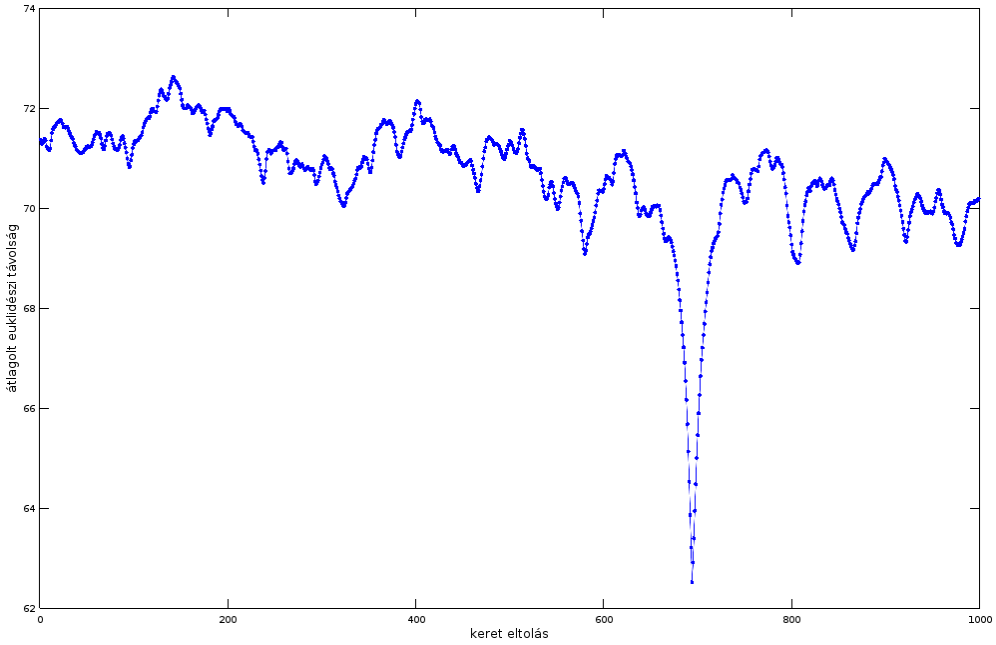
Két vektort összehasonlíthatunk úgy, hogy euklidészi távolságot számítunk közöttük. Az euklidészi távolság képlete:

$$d(x,y) = \left(\sum_{i=1}^{n} \left(|x_i - y_i|\right)^2\right)^{1/2}$$

ahol x és y két vektor, amelyeknek a távolságát számítjuk, n a tér dimenziószáma.

Esetünkben két hangmintát szeretnénk összehasonlítani. A két hangmintából MFCC lényegkiemeléssel két vektorsorozatot hozunk létre. A sorozatok indexe mindkét esetben azonos időbeli lépést jelent, ezért a két vektorsorozat soron következő elemei megfeleltethetők egymásnak. Ezt szem előtt tartva következőképpen számítjuk a két vektorsorozat között a különbözőség mértékét: „Egymás fölé” írjuk a két vektorsorozat elemeit, az így megfeleltetett vektor-párok között euklidészi távolságot számítunk, és ezek számtani közepét vesszük.

Az egymás fölé helyezést minden lehetséges eltolással elvégezzük, és az így kapott számsorozat azt jelenti, hogy az egyes keret-lépésenként számított időpontokban mennyire különbözik az azonosítani kívánt hangminta a referencia hangmintától. A találat ennek a számsorozatnak az abszolút minimumában lehet. Abszolút minimum természetesen minden referencia zeneszámmal fog adódni, hogy ez tényleg az azonosítani kívánt hangmintával való egyezést jelenti-e, azt egy megfelelően megválasztott küszöbértékkel való összevetéssel tudjuk eldönteni. A 2.2. ábra egy ilyen illesztés átlagolt euklidészi távolág eredményeit mutatja az eltolás függvényében. A 700. keretnél látható egy erőteljes minimum. Ez megfelel annak az esetnek, hogy a referencia hangminta 7 mp időponttól kezdve tartalmazza az azonosítandó hangmintát.



2.2. Ábra: MFCC illesztés eredménye keretenként

3 Audio ujjlenyomatos hangminta azonosítás

3.1 Audio ujjlenyomatról átalában

Az audió ujjlenyomat származtatásának és felhasználásának sok változata található meg a szakirodalomban. Általánosságban az audió ujjlenyomatot egy audió jel tartalmának tömörített egyedi azonosítójának tekinthetjük. Az audió ujjlenyomattal szemben támasztott követelmények általában az alábbiak:

- Pontosság (a vele való azonosítás ne okozzon sok fals negatív, illteve pozitív hibát)
- Robosztusság (tömörítési eljárások, zajforrások, harmonikus eltolás, frekvencia kiemelés, stb. által kialakított változatosság elleni védelem)
- Jó időfelbontás (néhány másodperctől egy teljes műsorszámig jellemezni lehessen vele)
- Biztonság (ne lehessen „becsapni” az azonosító algoritmust)
- Rugalmasság (bármilyen típusú audió jel – beszéd, rock, pop, klasszikus zene, zajok – jellemzésére legyen alkalmas, változó alkalmazási környezetben)
- Skálázhatóság (nagyszámú azonosítandó referencia hangforráshoz és/vagy valós idejű felhasználóhoz)
- Gyorsaság (az ujjlenyomat számításához, a referenciákkal való összehasonlításhoz, az új azonosítandó elem hozzáadásához, stb. szükséges erőforrások)

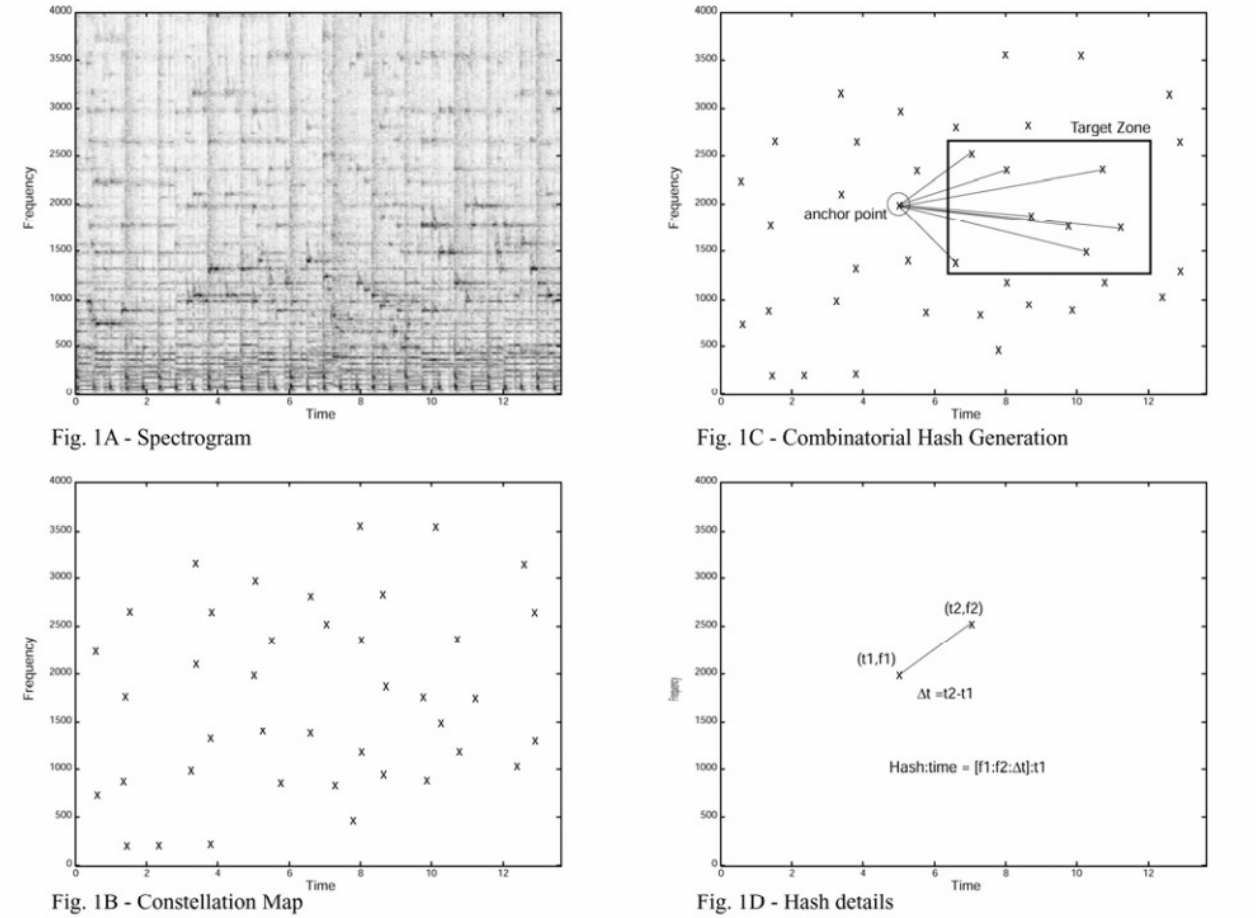
Az audió ujjlenyomat származtatásának és felhasználásának alapvető területei:

- meta-adatbázis létrehozása (audió adatbázisok tömör, jól indexálható jellemzésére)
- ismeretlen felvétel azonosítása a referenciákkal összehasonlítva (ill. annak megállapítása, hogy nem szerepel az adatbázisban)
- integritás ellenőrzés (annak vizsgálata, hogy a referencia hangfelvételt megváltoztatták-e)
- vízjelezés támogatása

Az audio ujjlenyomat tehát, mint az elnevezés is utal rá, egy olyan, nagyon tömör, de lehetőleg minél egyedibb jellemzése egy hangmintának, aminek segítségével elsősorban azonosítás jellegű eldöntendő kérdésekre adhatjuk meg a választ. Ezzel szemben a lényegkiemelés, bár használható azonosításra is, az azonosítási feladatok megoldásához szükségesnél sokkal több információt tartalmaz, és nem is elsősorban azonosításra találták ki. Kifejlesztésük elsődleges szempontja az, hogy minél teljesebb mértékben tartalmazza az emberi fül számára hasznos információt (a "lényeget") az eredeti hangmintából, a zavaró zaj hatását viszont küszöbölje ki. A hangminták lényegkiemelt változatával olyan, az azonosításnál sokkal bonyolultabb feladatok is megoldhatók, mint a beszéd felismerése, a beszélő felismerése, beszédben érzelem felismerése, zeneszámok műfaj vagy hangszer szerinti osztályozása, egyéb nem-beszéd jellegű hangok osztályozására.

3.2 Wang-féle audio ujjlenyomat

Talán a legismertebb, kimondottan zenei műsorszámok azonosítására szolgáló üzleti alkalmazás a Shazam. Ennek technológiai hátterét [1] alapján ismertetjük.



3.1. Ábra: A Shazam zeneszám azonosító rendszer spektrális audio ujjlenyomat képző módszere

A spektrális elemzés alapelvének illusztrálása a 3.1. ábrán látható. Az 1A részábra egy minta spektrogramot mutat be. Ezek szerint az elemzett frekvencia tartomány mindössze 4kHz-ig terjed. Ennek az a magyarázata, hogy a vezetékes vagy mobiltelefonon közvetített zene felismerését is meg kell oldani. A keret hossza 14 ms. A feldolgozás első lépése a spektrogramban található csúcsok keresése adaptív küszöblogika segítségével. Ennek eredményét láthatjuk az 1B részábrán. Ez már önmagában is jelentős adat redukciót biztosít. Az így létrejött pont mintázatot „konstelláció térkép”-nek (constellation map) nevezik a szerzők, mert gyakran emlékeztet az éjjeli

égboltra. A spektrogram-csúcsok értelme, jelentése inkább egy kottához, a zene dallamának leírásához áll közelebb, éppen ezért is alkalmas jól a zeneszámnak más zeneszámoktól megkülönböztető leírására.

A konstelláció térkép már jól illeszthető az ismeretlen jel és a referencia adatok elcsúsztatásával. Ha sok pont egyezik, akkor valószínű az azonos forrás. Ez a megoldás eléggé érzéketlen a kódolásból, a frekvencia kiemelésből és a lineáris torzításból adódó eltérésekre. Viszont ez az illesztés még egy elég lassú folyamat. Ezért úgynevezett horgonypontokat (anchor point) jelölnek ki (akár minden spektrális csúcspont lehet horgonypont) és ezekhez rendelnek egy elemzési területet (target zone). A horgonypontokat párba rendezik a hozzájuk tartozó elemzési területen található csúcsokkal. Ezeket a pontpárokat landmarkoknak is nevezik. Minden pár két frekvencia értékkel (a horgonypont és az elemzési területen található csúcs frekvenciája) és egy időtartam értékkel (a horgonypont és az elemzési területen találgató csúcs időbeli távolsága) jellemezhető. Ezt illusztrálja az 1C és 1D részábra. A frekvencia és az időkülönbség értékeket is 10 biten vannak ábrázolva, ez a két frekvenciából és egy időkülönbségből álló hármas képez egy 32 bites hasht. Minden referencia zeneszámra lefut a fenti algoritmus, és egy táblázatban eltárolódik a hash értéke (ami két csúcspont frekvenciájának, és a köztük lévő időkülönbségnek felel meg), az első csúcspont időcímkéje a hangfájl kezdetétől számítva, és a zeneszám azonosítója. A gyors kereshetőség érdekében hash kód szerint rendezve tárolódnak. Ezeket a táblázatba rendezett értékek képezik a zeneszámok audio ujjlenyomatát.

3.3 Zeneszám felismerés a Wang-féle módszerrel

A bejövő hangminta azonsítása a következőképpen történik: Az azonosítandó hangmintára is lefut ugyanaz az ujjlenyomat képző módszer, mint a referencia hangfájlokra, és létrejönnek a hash-kezdő időcímke párosok. Az egyező hash kódú bejegyzéseket kikeresi a rendszer az adatbázisból, és egy gyors algoritmussal megállapítja, hogy van-e elegendő számú, olyan páros, ahol a hash érték megegyezik, és az időbeli eltolások különbsége a referencia és az azonosítandó hanganyag között közel azonos. Ez az eltolás-különbség jelenti azt a referencia zeneszámban mért időpontot, ahonnan kezdve az azonosítandó hanganyag rögzítésre került.

Ez a zeneszám azonosítási módszer a célszerűen kigondolt, egyszerűen, gyorsan számítható audio ujjlenyomat elve, a nagy mennyiségű, hash szerint rendezett táblázatban tárolt adat és a gyors ujjlenyomat illesztő módszer felhasználása miatt igen gyors tud lenni. Egy kb. 20.000 zeneszámot tartalmazó adatbázisban való keresés futása egy PC-n 5-500 ms ideig tart. Ez a két nagyságrendnyi tartomány abból adódik, hogy ha nagyon rossz minőségű az azonsítandó hanganyag, akkor nagyobb mennyiségű ujjlenyomat számítódik, mint ha jó minőségű zeneszám-részletre számít a rendszer.

A fenti módszernek a mérésünkben használható Octaveban/Matlabban megírt implementációja egy leegyszerűsített változat, sebessége természetesen jóval elmarad a sok szempont alapján optimalizált ipari rendszerétől.

4 A rendszer teljesítményének kiértékelése

4.1 Osztályozás kiértékelése

A mérésünkben vizsgálandó eldöntendő kérdés, miszerint egy adott azonosítandó hangminta illeszkedik-e egy adott referencia hangmintára, egy egyszerű osztályozási probléma. Binárisnak nevezzük az osztályozási problémát, ha összesen 2 osztály között kell dönteni.

Esetünkben a 2 osztály:

- a teszt hangminta illeszkedik a referencia hangmintára (pozitív)

- a teszt hangminta nem illeszkedik a referencia hangmintára (negatív)

Annak tekintetében, hogy a vizsgált módszer mire döntött, és helyesen döntött-e, a következő eseteket különböztetjük meg:

- Igaz pozitív (szokásos angol elnevezéssel true positive, TP): A vizsgált módszer a „pozitív” osztályra döntött, és ez a döntése helyes volt

- Hamis pozitív (false positive, FP): A „pozitív” osztályra döntött, de helytelenül, a „negatív” lett volna a helyes döntés. Ezt 1-es típusú vagy elsőfajú hibának is nevezik.

- Igaz negatív (true negative, TN): A „negatív” osztályra döntött, helyesen.

- Hamis negatív (false negative, FN): A „negatív” osztályra döntött, helytelenül. Ezt 2-es típusú vagy másodfajú hibának is nevezik.

Általában az ilyen bináris osztályozási problémákat a következő mérőszámokkal szokás kiértékelni:

- Pontosság (Accuracy) = (TP+TN)/(TP+TN+FP+FN): a helyesen felismert adatok aránya

- Felidézés (Recall) = TP/(TP+FN): a pozitívak helyesen felismert aránya

- Precizitás (Precision) = TP/(TP+FP): a pozitívként felismertek hanyad része volt valóban pozitív

- F-mérték (F-measure) = 2*Precision*Recall/(Precision+Recall): harmonikus közepe a precison és recall értékeknek

4.2 Sebesség kiértékelése

Az olyan műveleteknél, amelyek gépidőigénye a bemenő adat mennyiségétől nagyjából lineárisan függ, és a bemenő adatok valamilyen időbeli adatból származnak, célszerű a művelet sebességét ennek a lineáris függésnek az arányszámával jellemezni. Például ilyen művelet esetünkben az MFCC számítás, mivel a bemenő hanganyag időbeli hosszával arányos a belőle készített keretek száma, és minden keretre ugyanannyi a számításigény. A számításhoz szükséges időnek és a bemeneti adat időbeli hosszának hányadosát real-time fectornak nevezzük:

$RTF = t_{számítás} / t_{adat}$

Tehát minél kisebb ez az érték, annál gyorsabb a feldolgozó algoritmusunk.

5 A méréshez használható scriptek és függvények

5.1 Octave függvények

A mérést nagyrészt GNU Octave környezetben kell elvégezni. Ez a Matlabhoz hasonló, matematikai számításokra készített interpreteres rendszer. A Matlab alapvető funkcióit valósítja meg, azzal kompatibilis módon. Használata ingyenes.

A mérés elvégzéséhez szükséges függvények rendelkezésre állnak egy könyvtárban,

vagy telepítve vannak Octave package-ként. Ezeknek a függvényeknek a használatáról következik egy rövid, nem minden részletre kiterjedő leírás. A függvényekről részletesebb leírás elérhető általában a „help ***függvénynév***” parancs kiadásával.

Az MFCC lényegkiemelő, valamint az audio ujjlenyomat képző és illesztő eszközök a http://labrosa.ee.columbia.edu/matlab/rastamat/ és a http://labrosa.ee.columbia.edu/matlab/fingerprint/ oldalakról származnak.

Y = wavread (FILENAME)

Betölt egy wav formátumú FILENAME nevű hangfájlt, a mintákat az Y vektorba teszi. Ha a WAV fájl több csatornás, Y mátrix lesz.

pl.: ***y_lvnp = wavread('mintak/lvnp.wav');***

PLAYER = audioplayer (Y, FS)

Létrehoz egy lejátszó objektumot, ami az Y vektorban adott hullámformát tudja lejátszani FS mintavételi frekvenciával.

pl.: ***player_lvnp=audioplayer(y_lvnp, 16000);***

play (PLAYER)

Elindít egy hanglejátszást a megadott, audioplayer() függvénnyel létrehozott objektummal. A mérésben akár ezt, akár külső hanglejátszó programot használhat a minták meghallgatására.

plot (X, Y, FMT)

Az aktuális ábra (figure) aktuális rész-ábrájára (subplot) egy diagramot rajzol. A kirajzolt pontok x és y koordinátái az X és Y vektor elemei, rendre. A fenti szintaxistól sok eltérő módon is hívható, lásd a helpet. Az X például elhagyható.

pl.: plot(e, '.-')

tic

Indít egy időmérőt.

VAL = toc

Visszaadja, vagy változónak való értékadás híján kiírja az időmérő indítása óta eltelt időt. Függvényhívások vagy nagyobb program részletek időigényét lehet vele megmérni.

pl.: tic;e=illeszt(mfcc_teljes_tiszta, mfcc_reszlet_mik);toc;

[S, F, T] = specgram (X, N, FS, WINDOW, OVERLAP)

Spektrogramot számít. A keretek hossza, ami egyben az FFT számítás pontszáma N. A keretek eltolási lépése OVERLAP minta, vagy ha ez nincs megadva, akkor N/2. FFT számítás előtt ablakozást is végez, az ablakozás típusa a WINDOW paraméterben adható meg, vagy ennek hiányában "Hanning" ablakot használ.

pl.: as_teljes_tiszta=abs(specgram(y_teljes_tiszta, 512));

specgram_plot(AS)

Megjeleníti a specgram függvénnyel kapott spektrum abszolútértékeként kézített amplitúdóspektrumot.

pl.: specgram_plot(as_teljes_tiszta)

[cepstra,aspectrum,pspectrum] = melfcc(samples, sr, varargin)

MFCC lényegkiemelést végez az adott hullámformán. Meg kell adni a mintavételi frekvenciát, amivel a hullámforma vektor készült. Ezen kívül még a lényegkiemelési folyamat sok paraméterét be lehet állítani, lásd a helpet. A 2. és 3. kimenet a szűrősor kimenet és a teljesítményspektrum, elhagyható, ha nincs rá szükségünk.

Figyelem! A kimenet transzponálva lesz az illeszt() függvénynek megfelelő a dimenzióit tekintve. Tehát így használja, ahogy a példában is látszik, transzponálással a sor végén:

pl.: mfcc_teljes_tiszta=melfcc (y_teljes_tiszta, 16000)'

E = illeszt(REF, TEST)

Végigtolja a TEST vektort a REF vektoron az 1. dimenziója szerint, és minden illesztésre számít egy átlagolt euklidészi távolságot. Ha a TEST és REF 2. dimenziója > 1, tehát mátrixok (mint pl. amplitúdó spektrum vagy MFCC adatoknál), akkor az euklidészi távolság számítása vektorok között történik.

pl.: e_tzm=illeszt(mfcc_teljes_tiszta, mfcc_reszlet_zaj)

I = etcsucs(ET);

Az illeszt() függvénnyel készült euklidészi távolság vektorban elég meredek csúcsot (minimumot) keres.

A visszatérési érték a megtalált minimumhely indexe az ET tömbben. Ha nincs találat, akkor 0-t ad vissza.

pl.: etcsucs(e_tzm)

clear_hashtable

Létrehozza és alaphelyzetbe állítja (törli) az ujjlenyomatos kereséshez használt hash táblát.

[N,T] = add_tracks(N,ID)

Hozzáadja az N cell-arrayben adott nevű wav fájlokat az ujjlenyomatos kereső rendszer referencia adatbázisához. Elvégzi rajtuk a teljes ujjlenyomat készítő folyamatot, és a kapott hasheket tárolja a hash-táblában. A 2. paraméterrel meg lehet adni explicite a fájlok hash-táblabeli azonosítóját, de el is lehet hagyni. Lehet hullámforma vektorral is hívni, lásd a helpet. A visszaadott értékek a hozzáadott trackek száma és a hívás időtartama mp-ben.

pl.:

tks{1}='zeneszamok/lvnp.wav';

tks{2}='lvnp_GSM.wav';

clear_hashtable;

add_tracks(tks);

[R,L] = match_query(D,SR,IX)

Ujjlenyomatos összehasonlítást végez. A D vektorban adott hullámformára végrehajtja az

ujjlenyomatozást, majd a hash táblában egyezéseket keres. Az SR változában a mintavételi frekvenciát kell megadni. Az R változóban adja vissza a találatokat, a L változóban az IX indexű referencia fájlhoz tartozó illesztett landmarkokat.

Az R mátrix sorai egy-egy referencia hangmintához tartozó landmark illeszkedéseknek felelnek meg. Az oszlopok jelentése rendre:

- referencia hangminta index,

- illeszkedő landmarkok száma,

- illeszkedő tesztanyag eltolása 32 ms-os egységekben mérve.

Lehet kérni részletes eredményt az IX változó megadásával, ekkor az L mátrix az IX indexű referencia hangmintával illeszkedő landmarkokat adja meg, soronként egyet-egyet. Az oszlopok jelentése rendre:

- az illeszkedő landmarkot képező csúcspár 1. tagjának időbeli helye a referencia hanganyagban, 32 ms-os egységekben

- a csúcspár 1. tagjának frekvenciája 15.625 Hz-es egységekben. (8000 Hz-es, 512 pontos FFT lépésközeként adódik ez az egység)

- a csúcspár 2. tagjának frekvenciája, 15.625 Hz-es egységekben

- a csúcspár 2 tagjának időbeli távolsága, 32 ms-os egységekben

- a teszt hanganyagnk az illeszkedést eredményező eltolása a referencia hanganyaghoz képest, 32 ms egységekben

Ha a landmark-illeszkedések részleteire nem vagyunk kíváncsiak, indítható kevesebb paraméterrel is:

pl.: match_query(y_reszlet_GSM,16000)

[DM,SRO,TK,T] = illustrate_match(DQ,SR,FL,IX)

Spektrogramon ábrázolja egy teszt és referencia hanganyagban talált landmarkokat, kiemelve az illeszkedőket. Futtatja a match_query-t is a rajzolás előtt.

Két spektrogramot rajzol, lásd a 4.1. ábrát. A felső a tesztanyagot ábrázolja, az alsó a referencia anyagnak a tesztanyaggal megegyező hosszú részét, az illeszkedésnek megfelelő eltolással. Egy landmark, vagyis egy spektrogram-csúcspár két körrel, és az őket összekötő vonallal van ábrázolva. Ezek sokszor összefüggő töröttvonalat vagy elágazó gráfszerű ábrát alkotnak, hiszen egy spektrogram-csúcs több csúcspárban is szerepelhet. Zölddel vannak jelölve azok, amelyek a referenciában és a tesztanyagban is szerepelnek, azonos helyen, azonos eltolással, pirossal a többi, nem illeszkedő.

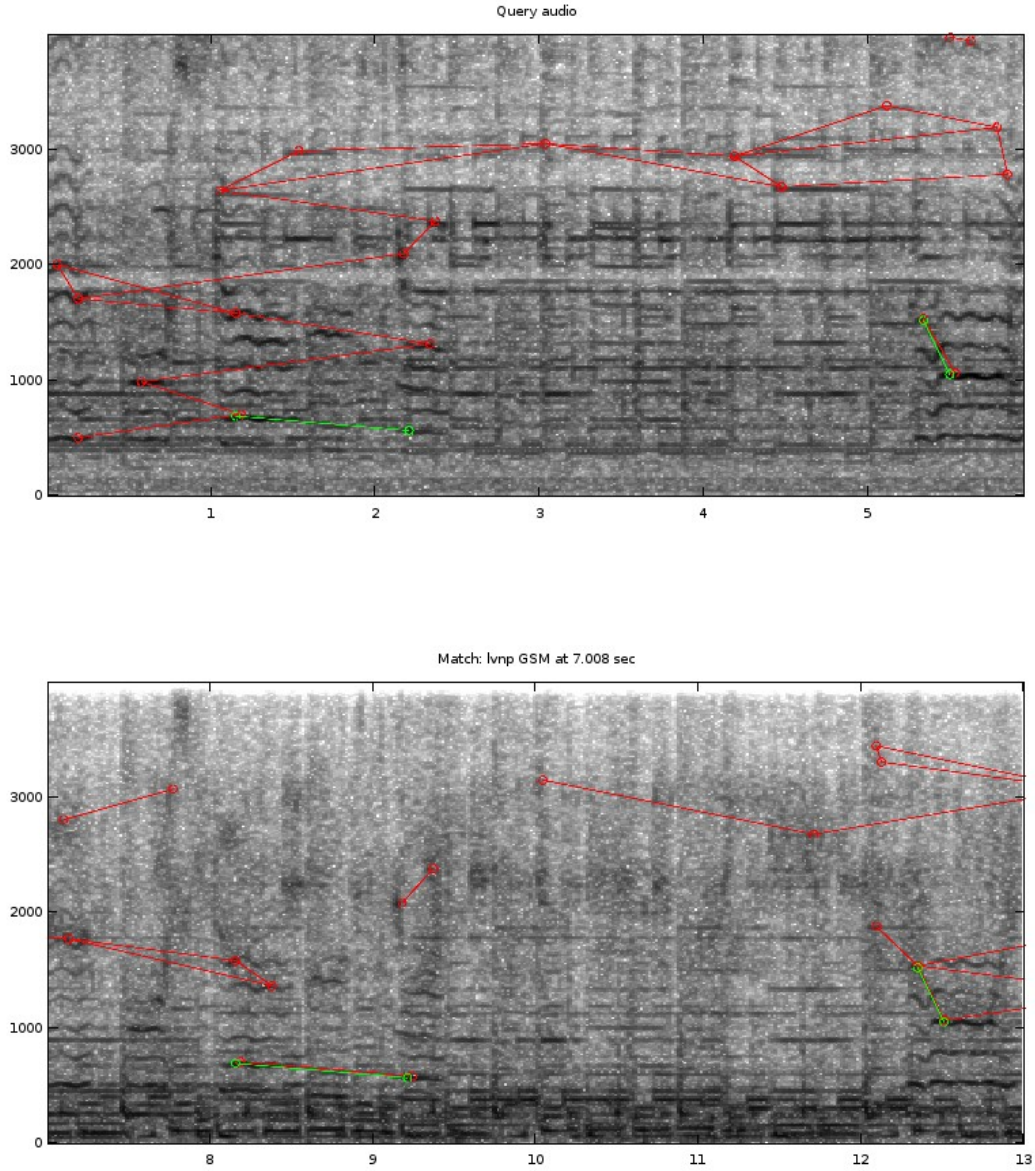
A bemenő paraméterek: DQ: teszt hanganyag hullámformája, SR: mintavételi frekvencia, FL: adatbázisban lévő hanganyagok neveinek cell-array-e, IX: megjelenítendő illeszkedés indexe

Kimenő változók: DM: teszt hanganyaggal illeszkedő referencia hanganyag részlet, SRO: mintavételi frekvencia, TK: illeszkedő referencia hanganyag azonosítója, T: illeszkedés időpontja a referencia hanganyagban 32 ms-os egységekben

pl.:

illustrate_match(y_reszlet_mik,16000,tks);

colormap(1-gray)



5.1. Ábra: Illeszkedés illusztrálása spektrogramon

X = csvread (FILENAME)

Beolvas egy mátrixba egy CSV (comma-separated-values = vesszővel elválasztott értékek) formátumú szöveges fájlt.

Pl.: kt=csvread ('kereszt_teszt_2000.csv');

#{ és #}

Ezek nem függvények, hanem a többsoros kommentezés jelölése. Kényelmes a mérés megoldása közben, ha a már megoldott részfeladatot ezzel a jelöléssel bezárójelezzük, így megmarad a .m fájlban a már megírt kód, de amikor újra futtatjuk a scriptet, már nem fut le újra.

Figyelem! Ezek a jelölések külön sorba írandók, vagyis nem lehet más is abban a sorban

ahol #{ vagy #} van.

Pl.:

#{

 # 1. a) feladat megoldása

x=y+1;

 # stb.

#}

 # 1. b) feladat megoldása

x=y+2;

disp (X)

Kiírja a paraméterének értékét. Változót vagy idézőjelek közé tett stringet is lehet neki adni. Hasznos lehet az eredmények megjelenítésekor az adatok felcímkézésére.

Pl.:

t=toc;

disp ("A futási idő:");

disp (t);

5.2 SOX scriptek

A SOX egy ingyenesen használható, parancssoros hangfájl-feldolgozó eszköz. Igen széles körű funkció készlettel rendelkezik, a készítőinek meghatározása szerint „a hangfeldolgozás svájci bicskája”. A mérésben jellemzően ezt a programot célszerű használni a tesztelendő hangmintán végzendő különféle torzítások, minőség rontások mesterséges létrehozására.

A SOX program funkcióinak teljes körű ismerete nem része a mérésre való felkészülési követelménynek. A hangminták feldolgozását, torzítását előre elkészített egyszerű (a SOX-ot megfelelő paraméterezéssel futtató) scriptekkel kell elvégezni, ezek ismertetése következik alább.

gain.sh

Paraméterek:

1.: bemenő wav fájl neve

2.: erősítési tényező dB-ben

3.: kimenő wav fájl neve

A bemenő hullámformát erősíti, vagyis szorozza a decibelben megadott erősítési tényezőnek megfelelő értékkel. Túlvezérlés jellegű torzítást lehet előidézni vele, ha olyan magasra állítjuk az erősítést, hogy a kimenet nem fér bele a számábrázolási tartományba.

Pl.: sox_scriptek/gain.sh zeneszamok/lvnp.wav 20 lvnp_tulvezerelt.wav

highpass.sh

Paraméterek:

1.: bemenő wav fájl neve

2.: vágási frekvencia

3.: kimenő wav fájl neve

Felüláteresztő szűrővel szűri a bemenő hullámformát, a megadott vágási frekvenciával.

Pl.: sox_scriptek/highpass.sh zeneszamok/lvnp.wav 4000 lvnp_highpass.wav

gsm_kodolo.sh

Paraméterek:

1.: bemenő wav fájl neve

2.: kimenő wav fájl neve

GSM kódolást végez a bemenő hangfájlon. Ez adatvesztéses tömörítést, minőségromlást jelent. Majd visszaalakítja 16 kHz, 16 bit PCM formátumba, tehát a kimenő fájl formátuma meg fog egyezni a bemenő fájléval.

Pl.: sox_scriptek/gsm_kodolo.sh zeneszamok/lvnp.wav lvnp_gsm.wav

kever.sh

Paraméterek:

1.: bemenő wav fájl neve

2.: hozzákeverendő wav fájl neve

3.: kimenő wav fájl neve

Összekever, vagyis mintánként összead két hullámformát. A kimenő fájl hossza az 1. paraméterben adott fájléval fog megegyezni. Tehát a 2. paraméterbe kell tenni a hozzákeverendő zajt.

Megjegyzés: Ha a zaj hangerejét módosítani szeretnénk, futtassuk rá először a gain.sh-t.

Pl.: sox_scriptek/kever.sh zeneszamok/lvnp.wav zajok/SIGNAL019-20kHz-2min_16k.wav lvnp_beszed.wav

mp3_kodolo.sh

Paraméterek:

1.: bemenő wav fájl neve

2.: kimenő wav fájl neve

MP3 kódolást végez a bemenő hangfájlon. Ez adatvesztéses tömörítést, minőségromlást jelent. A szokásos MP3 minőséghez képest sokkal erőteljesebb tömörítés van beállítva, a látványosan rossz végeredmény érdekében. Ezután visszaalakítja 16 kHz, 16 bit PCM formátumba, tehát a kimenő fájl formátuma meg fog egyezni a bemenő fájléval.

Pl.: sox_scriptek/mp3_kodolo.sh zeneszamok/lvnp.wav lvnp_mp3.wav

speed.sh

Paraméterek:

1.: bemenő wav fájl neve

2.: speed paraméter

3.: kimenő wav fájl neve

Sebesség módosítást végez a megadott hangfájlon, az adott arányban. Ez az adott hullámforma nyújtását vagy szűkítését jelenti, tehát a tempóval együtt a hangmagasság is változni fog. A speed paraméter 1-nél nagyobb értékére gyorsulás, 1-nél kisebb értékre lassulás történik.

Pl.: sox_scriptek/speed.sh zeneszamok/lvnp.wav 1.3 lvnp_magas.wav

tempo.sh

Paraméterek:

1.: bemenő wav fájl neve

2.: tempo paraméter

3.: kimenő wav fájl neve

Tempó módosítást végez a megadott hangfájlon, az adott arányban. Ez olyan feldolgozást jelent, aminek következtében csak a zene tempója változik, a hangmagasság nem. A tempo paraméter 1-nél nagyobb értékére gyorsulás, 1-nél kisebb értékre lassulás történik.

Pl.: sox_scriptek/tempo.sh zeneszamok/lvnp.wav 1.3 lvnp_gyors.wav

feherzaj.sh

Paraméterek:

1.: bemenő wav fájl neve

2.: a zaj amplitúdója

3.: kimenő wav fájl neve

Fehérzajt generál adott amplitúdóval, és hozzáadja a bemenő hullámformához.

Pl.: sox_scriptek/feherzaj.sh zeneszamok/lvnp.wav 0.3 lvnp_feher.wav

trim.sh

Paraméterek:

1.: bemenő wav fájl neve

2.: kivágás kezdete mp-ben

3.: kivágás hossza mp-ben

4.: kimenő wav fájl neve

Kivág egy darabot a megadott hangfájlból.

Pl.: sox_scriptek/trim.sh zeneszamok/lvnp.wav 7 6 lvnp_reszlet.wav

6 Irodalomjegyzék

[1] Wang, Avery, et al., „An Industrial Strength Audio Search Algorithm”, In: ISMIR. 2003. p. 7-13 (shazam)

[2] P. Cano, E.Batlle, T. Kalker, J. Haitsma, „A Review of Audio Fingerprinting”, Journal of VLSI Signal Processing 41, 2005, pp. 271–284

[3] H. B. Kekre, Nikita Bhandari, Nisha Nair, Purnima Padmanabhan, Shravya Bhandari, „A Review of Audio Fingerprinting and Comparison of Algorithms” International Journal of Computer Applications, Volume 70 - Number 13, pp. 24-30

[4] Pang-Ning Tan, Michael Steinbach, Vipin Kumar „Bevezetés az adatbányászatba”, TAMOP 4.2.5 Book Database, http://www.tankonyvtar.hu/en/tartalom/tamop425/0046_adatbanyaszat/ch05s07.html