Új kulcsszavak és dokumentumok keresése az interneten

Szakdolgozat

Budapest, 2003. Június 4.

Molnár György
Programtervező matematikus
V. évfolyam

Témavezető: Dr. Lőrincz András

Köszönetnyilvánítás

Ezúton szeretném megköszönni témavezetőmnek Dr. Lőrincz Andrásnak – az ELTE NIPG vezetőjének –, továbbá az ELTE Neurális Információfeldolgozási Csoport tagjainak, különösen Palotai Zsoltnak, hogy ötleteikkel, javaslataikkal nagymértékben segítették munkámat.

1 Tartalomjegyzék

2 Ábrajegyzék

3 Új dokumentumok az interneten

A legnagyobb információforrás napjainkban az internet. A dokumentumok száma elérheti a 10 milliárdot. A dokumentumok folyamatosan változásának a mértéke szintén óriási. Ez az állandó növekedés komoly kihívást jelent az új információk keresésében.
A WWW skálamentes struktúrával rendelkezik: egy gráf egy skálamentes hálózat, ha a inmenő (vagy bejövő vagy mindkettő) élek száma hatvány-eloszlást követi ($P(k) \propto k^{-\gamma}$, ahol k egész, $P(k)$ annak a valószinűsége, hogy a pontnak k bejövő (vagy kimenő, vagy mindkettő) éle van, és $\gamma>0$). A közvetlen következménye a skála-mentes tulajdonságnak, hogy számos URL vagy összekapcsolt URL-halmaz létezik, amelynek sok bejövő éle van. Intelligens web-crawlerek könnyen csapdába eshetnek ilyen csomópontok mellett.
Kifejlesztettünk egy újszerű mesterséges élet (Alife) eljárást intelligens egyedekkel (ügynökök), melyek képesek detektálni az un. 'breaking news' tipusú híreket hatalmas WWW domain-eken. Azért fordultunk az Alife technológiához, hogy hatékonyan tudjuk felosztani a munkaerőt és közben a részlegek közt minimalizáljuk a kommunikációt. Az ügynökök megerősítéses tanulást használva számítják a hosszútávú összegzett jutalmakat és ennek segítségével keresnek. A számítások függvény-approximációt és temporális különbség tanulást (temporal difference learning) használnak. A bemenetét a függvény-approximátornak egy *PrTFIDF* (Probabilistic term-frequency inverse term frequency) osztályozó biztosítja, amely egy jól ismert szöveg-csoportosító eljárást használ.

3.1 A kereső eljárás

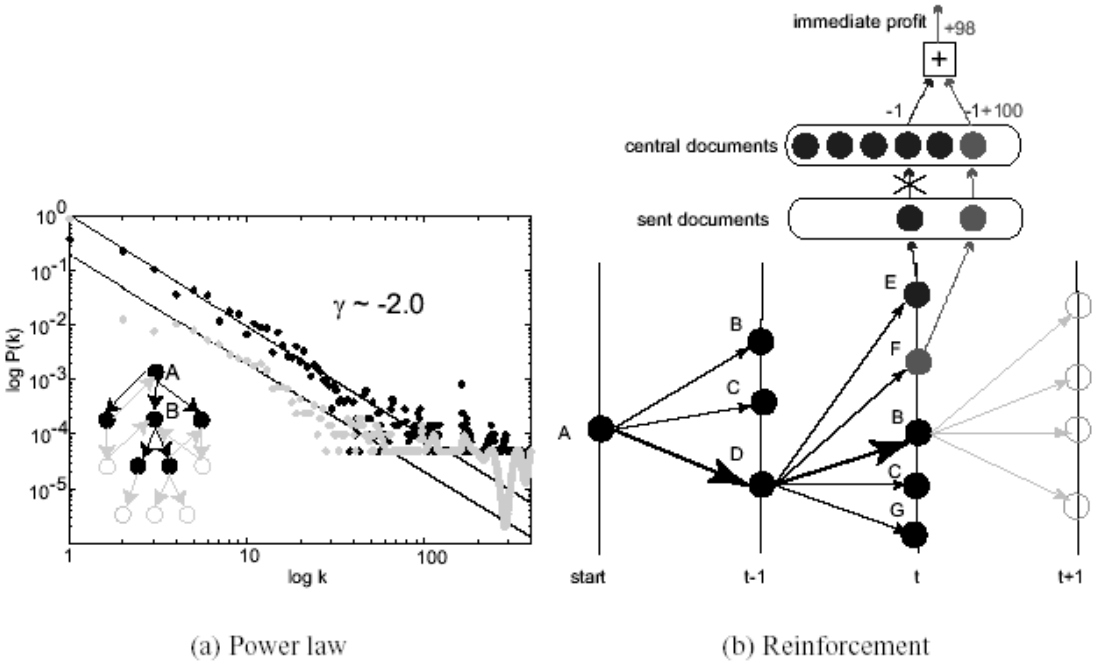
Az eljárásban az egyedek *kutatók*, benépesítenek egy folyamatosan változó világot, ahol a felbukkanó új erőforrások aránya limitált.

Környezet. A kisérleteink domain-je (az egyik legnagyobbn hír-portál), mint a legtöbb WWW domain skála-mentes volt a méréseink szerint (ld. 1(a) ábra). Mind a kimenő, mind a bejövő linkek eloszlása hatvány-eloszlás volt. A bemenő halmaz (inset?) megmutatja a linkeket és dokumentumokat, melyeket a kutatók megvizsgálnak. Az eloszlás megfelel a kutatók által megvizsgált linkeknek. A hírkutatók meglátogatják az 'A' URL-t, letöltik a még nem látogatott környezetét. A letöltést egy döntés követi, aztán a 'B' URL következik, a letöltés megkezdődik, és így tovább. Az 'A', 'B', 'C', … sorozatot útnak hívjuk.

A kutatók. A kutatóknak két komponenű hosszútávú memóriája van. Az első komponens a döntéshozatalt szolgálja, és meghatározza a viselkedését a kutatónak. A második komponens egy véges méretű memória, amely eljárásokat tartalmaz. Ez a 'procedúrális memória' tárolja, hogy hogyan kell a csapdákból kikerülni. Itt az eljárás egyszerűen váltani egy másik URL-re, egy másik kezdőpontra. A kutató egy adott lépést keres. A lépések a kutató viselkedéséhez igazodik. Miután megtette a lépéseket, az eljárás, amelyet véletlenszerűen választ a procedúrális memóriából, megoldja a csapdázódás problémáját.
Minden meglátogatott oldalnak a kutató letölti a szomszédos dokumentumait, és megállapítja, hogy egy dokumentumnak van-e időpecsétje az aktuális dátummal. Ha van,

akkor a dokumentumot elküldi a központnak. Ezek a dokumentumok az 'elküldött' jelzőt kapják.

A jutalmazó rendszer: A kutatók 'élelem' után kutatnak, amik jelen esetben az új hírek. A központi megerősítő egység pozitív és negatív jutalmat oszt. Pozitív jutalmat csak a hír első beküldője kap, ha a hír időpecsétje nem öregebb, mint 1 nap. Ekkor c_+ jutalmat oszt (ami 100 tetszőleges egységben (a.u.)). Minden további kutató, aki ugyanezt a dokumentumot küldi c_- büntetést kap (ami –1-el egyenlő a.u.). Az azonnali profit a jutalom és a lépés költségének különbsége. Az 1(b) ábra megmutatja egy példa útvonal megerősítését: a start időpillanatban (t-2), az ügynök az ***A*** URL-en van, melynek szomszédait – ***B***, ***C*** és ***D*** – letölti. A ***D*** URL-re lét következőleg. Az ***E***, ***F*** és ***G*** URL-t letölti. A ***G*** dokumentumnak elavult az tidőpecsétje. Az ***E***-t és az ***F***-et elküldi a központnak. Az ***F*** új a centernek, ezért jutalmat kap a kutató. Ebben a körben 98 jutalmat kap a kutató. A kutató nyilvántart egy listát a látogatott URL-ek szomszédairól, ezeket *határ*nak hívja. Csak a határon lévő URL-eket látogathatja meg.



(a) Power law (b) Reinforcement

1. ábra Skála-mentes tulajdonsága egy internet domain-nek és a jutalmazó rendszer

(a): Log-log skálán a linkek eloszlása az összes URL-en a vizsgálat alatt. Vizszintes tengely: az élek száma (log k). Függőleges tengely: Az élek számának relatív frekvenciája különböző URL-eken (log P(k)). Fekete (szürke) pontok: Bejövő (kimenő) élei az URL-eknek. A meredeksége a függőleges vonalnal 2 ±0.3. Belső ábra: a letöltés menete. Fekete (szürke) linkek: benne van (nincs benne) az adatbázisban. Kitöltött (üres) körök: benne van (nincs benne) az adatbázisban **(b):** Központi megerősítő rendszer. Üres

(kitöltött) körök: (nem) új dokumentumok. Pozitív (negatív) számok: jutalom és profit (költség) Függőleges szaggatott vonalak: egymást időben követő lépések. Pontok a (t+k)-adik szaggatott vonalon: az elérhető dokumentumok a (t+k-1)-edik lépésben.

Hosszú távon számított profit (Long term cummulated profit, LTP): Az azonnali profit egy rövidlátó jellemzése egy URL-nek. A kutatók intelligensen viselkednek, ha azt a politikát követik, amely maximalizálja a várható LTP-t az azonnali profit helyett. A politika és a profit-számítás összekapcsolt fogalmak, hiszen a profit számítás meghatározza a politikát, a politika pedig meghatározza a választásokat, tehát az LTP-t is. Itt a választás a mohó LTP politikára épül. A kutató meglátogat egy URL-t, ami a határhoz tartozik és a legnagyobb számított LTP-je van. A meglátogatott URL-ek egy utat formálnak, és minden út maximum 100 lépés hosszú.
Minden kutatónak k (=50) dimenziós PrTFIDF szövegosztályozója van, amit egy korábban letöltött geocities adatbázis alapján generáltunk. 50 csoportot generáltunk a Boley csoportosító algoritmussal a letöltött dokumentumokból. A PrTFIDF osztályozó ezeken a csoportokon lett betanítva, plusz egy olyanon, amely általános szöveget jellemez az internetről. A PrTFIDF kimenetét nem lineárisan leképezzük a [-1,+1] intervallumra egy hiperbolikus tangens függvénnyel. Az osztályozó segítségével a szövegeket leképezzük egy kis dimenziós reprezentációra. Amikor egy kutató meglátogatja ***A*** URL-t, a dokumentumot d_a-nak, az őt reprezentáló vektort s_a=($s_a(1)$, …, $s_a(k)$)-nak jelöljük. A (k+1). kimenetet elhagyjuk.
Lineáris approximátort használunk az LTP becsléséhez. Használja a *k* paramétert és a súlyvektort w = (w(1),…,w(k)). A dokumentum LTP-je d_a a skalár szorzata s_a és w vektoroknak: $L(da_n) = \sum_{i=1}^{k} w(i)s_a(i)$. Ez a súlyvektor az első komponense a kutatók hosszútávú memóriájának. A súlyvektorokat megtámogattuk TDL-el (temporal difference learning): jelölje a következő meglátogatandó dokumentumot d_n, az osztályozó kimenetét s_n, és a becsült LTP értéke a dokumentumnak $L(d_n) = \sum_{i=1}^{k} w(i)s_n(i)$. Tegyünk fel, hogy elhagyva a d_a dokumentumot, és megérkezve a következő dokumentumra, az azonnali profit r_n. A becslésünk tökéletes, ha $L(d_a) = L(d_n)+r_n$. A jövőbeni profitot általában figyelmen kívül hagyjuk ilyen becslésekben: $L(d_a) = \gamma L(d_n)+r_n$, ahol $0<\gamma<1$. Ekkor a hibaértéke a becslésnek:

$$\delta(a,n)=r_n+\gamma L(d_n)-L(d_a).$$

A szimuláció alatt végig γ=0.9-et használtunk. Minden $d_a \rightarrow d_n$ lépésnél az érték függvény súlyait finomhangoltuk, hogy minimalizáljuk az értékbecslés hibáját a látogatott dokumentumnak. A becslés hibáját használtuk, hogy javítsuk a paramétereket: az i. komponensét a súlyvektornak w_i-t a következőképpen javítottuk:

$$\Delta w_i = \alpha\delta(a,n)s_a(i)$$

ahol α=0.1, és i=1,..,k.

URL listák és döntések. A kutatók hosszú távú memóriájának második komponense, a procedúrális memória, a *weblog*, ami egy URL-lista, legfeljebb 100 elemmel. A kezdőpontok a lista első 10 eleme. Az út elején a kutató véletlenszerűen választ egyet a

kezdőpontok közül, és meglátogatja azt. Ha egy útat a kutató befejez, akkor választ egy új kezdőpontot.
Az ***A*** URL-nek az összegzett profitja az azonnali profitok összege, amit azon az úton gyüjtött, miután meglátogatta az ***A***-t. Utalva az ***A*** URL összegzett profitjára, $R_{path}(A)$ amikor az út befejeződött, az ***A*** URL értéke, utalva a $V(A)$-ra, a következőképpen számoljuk:

$$V_{new}(A) = (1 - \beta)V_{old}(A) + \beta R_{path}(A)$$

ahol β értékét 0.3-ra állítottuk. Ha az ***A*** URL-nek nem volt ezt megelőzően értéke, akkor $V_{new}(A)$ értéke egyenlő lesz $R_{path}(A)$ értékével. Ezeket az értékeket a weblog-ok frissítésére használjuk minden út után. Az URL-eket csökkenő sorrendbe rendezzük, majd a századiktól kezdve levágjuk. A megmaradt 100 alkottja az új weblogot.
A kutatók fenntartanak két rövid távú memóriát is minden út alatt. Az egyik a már meglátogatott oldalak listáját tartalmazza, hogy elkerülhesse a hurkokat. A másik a határ, azon oldalak listája, amelyek közvetlenül elérhetőek a látogatott oldalakról, beleértve magukat a látogatott oldalakat is. A kutató a következő lépését ebből a listából választja. Ha a lista üres, vagy az út hossza eléri a 100 lépést, akkor ez az út befejeződik.

Többszörözés biparticionálással és kihalással. Minden kutatónak 100 az értéke kezdetben. Ez az érték csökken 0.05-tel minden egyes központnak elküldött dokumentummal és nő 1-el, ha az elküldött dokumentumot jutalmazta. Ha a kutató értéke eléri a 200-at, a kutató többszöröződik biparticionálással, és mindkét rész értéke 100 lesz. Az eredeti weblog-ot véletlenszerűen két egyenlő részre osztjuk és odaadjuk a két utódnak a szülő súlyvektorával együtt. Másrészről, ha a kutató pontja eléri a nullát, akkor kihal.

Kutató periódus. A kutatók szekvenciálisan futnak egy előírt sorrendben, nagyjából egyenlő időintervallumban 1 PC-n (az utolsó látogatott URL analízisét (környezet letöltése, LTP becslés) még befejezheti). A kutató periódus az az időintervallum, amíg a kutatók egyszer lefut. A befejezetlen utat folytatja a következő futáskor.

3.2 Összegzés

Új internet-oldalak folyamatosan bukkantak fel az általunk vizsgált hír-portálon. A hír-kutatók populációjára tekinthetük úgy, mint egy gyorsan adaptálódó, önfejlesztő hírdetektorra. A hatékonysága és a sebessége az új hírek keresésének nő. Ez annak a ténynek az ellenében következik be, hogy a környezet struktúrája gyorsan változik. Időben az újonnan felfedezett hírek száma körülbelül konstans volt. Sok drasztikus változás érte a hír-detektor, ami folyamatosan újraépítette magát és növelte a monitorozás hatékonyságát.
Összegzésképpen, ez az eljárás demonstrálja, hogy evolúciós algoritmusok hatékonyan működhetnek skála-mentes hálózatokon. Hasonló eredményt várunk az internetes keresés különböző problémáiban is. Habár, mi úgy hisszük, hogy a sebesség és az osztott munka hatékonysága az Internet magas klaszterezettségéből és skála-mentes kisvilág struktúrájából adódik, a terep tele van csapdákkal. Ezek a csapdák kutaó-mezőkké váltak a két komponensű hosszú távú memóriát használva, (a) az igéretes kezdőpontok listája, amit mi procedúrális memóriának hívunk és (b) a hosszútávú profit becslő. Mindkét komponens szükséges. Az (a) nélkül a kutató nem tudna szabadulni a csapdából, a (b)

nélkül pedig nem tudná a munkát nagy hatékonysággal megosztani. A kutató populáció evolúciója gyorsan megy végbe, lehetővé téve, hogy sikeresen adaptálódjon a gyorsan változó világhoz. Ezeket a vonzó tulajdonságokat anélkül érik el, hogy direkt kommunikálnának a kutatók egymás közt, ami nagy előny kommunikációs hálózatoknál.

4 Témakör-térképek

Napjainkban az interneten naponta jelennek meg új dokumentumok, de ember számára kezelhetetlen mennyiségben. A legfőbb problémák:

1. A növekedés mértéke óriási
2. Nem az információ, hanem a frissessége a lényeges kérdés
3. Az információ sok esetben speciális. Alapvető dolgok sokszor nem érthetők meg speciális szótár nélkül.
4. A szótár jelenthet szószedetet, vagy szómagyarázatot (thesaurus). Ezek kiegészítő segédeszközök.
    a. az alapelem lesz a szószedet, pl. a szavak (term) magyarázata
    b. a fejlettebb elem a szómagyarázat (thesaurus)

A megfelelő objektum ebben az esetben egy fogalmi gráf, aminek építőelemei fogalmak és fogalmi kapcsolatok.
Nem csoda, hogy különböző kutatások indultak ezekben a témákban, mint például:

1. szemantikus hálók
2. témakör térképek (topic maps)

A témakör térképek például a hangsúlyt a témákra, asszociációkra és az előfordulásokra helyezik. A témakör térképek XML formája, mint a HyTM és az XTM nemrégiben lett kifejlesztve. Bár ezek a technológiák fontosak lehetnek, erős a kétség, hogy képesek lennének "meghódítani a világot". A benyomás ugyanis az, hogy senki sem tudná igazán rendesen megismerni őket, mielőtt elavulnának.
Nézzük, mit mondanak a témakör térképekről:
„... A témakör térképek születése a korali 90-es évekre tehető, amikor a későbbiekben Davenport Group néven megismert csoport megbeszélte a módjait az elektronikus dokumentumok cseréjének. A csoport elkezdte kifejleszteni a DocBook-ot ([DocBook 1999]), a legszélesebb körben elterjedt DTD-t (Document Type Definition) SGML és XML dokumentumok megosztására. A Davenport Group egyik nagy problémája az volt, hogyan egyesítsen különböző dokumentumok indexeit, és amit kitaláltak az a következő volt:

> Az indexek, amennyiben konzisztenek, hasonlóak az általuk indexelt dokumentumokban megtalálható tudás struktúrájának a modelljéhez. De a modell implicit, és sehol sem található. Ha egy ilyen modell formálisan megalkotható lenne, akkor nagyban megkönnyítené a modellezett indexek egyesítését. „

Úgy gondolom, ez a feljegyzés nagyon is igaz. A témakör térkép nagyon használható segédeszköz egy jól megalapozott témában: akkor, ha tudjuk mit keresünk és meg tudjuk találni. Ebben az esetben egy témakör térkép hatásos segítséget nyújt az anyagok gyors és könnyű áttekintéséhez. Nincs okunk rá, hogy azt gondoljuk, hogy új vagy még meg nem talált információk témakör-térképe minden esetben hasznos lehet.

4.1 Áttekintés

A számítástechnika és a telekommunikáció óriási adatbázisokat képes generálni. A legtöbb esetben ez több adatbázis összekapcsolásából születik. Másik példa az Internet maga. Nagyon fontos, hogy megértsük az adatbázis struktúráját és az alkalmazását a felfedezett struktúrábasn alkalmazásokban.

Legyen P adathalmaz adott. Az adathalmaz elemei legyenek pontok. A csoportosítás a P-beli pontok szétválasztása. A szétválasztásban hasonlósági mértéket használunk. Ez a mérték számokat rendel bármely pontpárhoz az adatbázisban. Legyen P például wbelapok halmaza. A hasonlósági mérték két lap közt lehet azon szavak száma, amelyek mindkét oldalban előfordulnak. Ezután csoportosítani megfelel a lapok téma szerinti felosztásnak. A csoportosítást hierarchikusnak nevezzük, ha először a P elemeit nagy csoportokra bontjuk, majd a csoportokat kisebbekre, és így tovább. Például a lapokat csoportosíthatjuk téma szerint zene, sport, politika, stb, azután a zenei csoportot tovább bonthatjuk, klasszikus zene, jazz, stb, a sportot tovább bonthatjuk futball, úszás, stb. A hierarchikus csoportosítás szerepet játszik a struktúra megtanulásában és az információ megtalálásában a struktúra alapján.

Az érdekes része a feladatnak, hogy hogyan és hány csoportra bontsuk a halmazt. Ha a csoportosítás azért készül, hogy segítse az emberi felhasználót, akkor a csportok száma egy szinten nem lehet több, mint 10.

A csoportosítás egy nagyjából megvizsgált terület. Sokféle halmaz és hasonlósági mértéket megvizsgáltak. Az egyik megközelítés lineáris algebrai megközelítés. Itt a pontok a P térben a_1, a_2, …, a_n attribútumaikkal jellemezzük. Ezeket n dimenziós vektoroknak tekintjük. A hasonlóságot ezután mérhetjük a szokásos módon, mint a pontok távolsága, vagy a vektorok által bezárt szög cosinusza.

Gráf-elméleti megközelítést használva. A P pontjai megfelelnek a G(V,E) gráf pontjainak, ahol V a pontok, E az élek halmaza. Ebben az esetben P=V. A hasonlóság az élek segítségével számítható. Lehet például a legrövidebb út két pont közt a hasonlósági mérőszám. Web-oldalak esetében az élek a lapok hyperlinkjeinek felelnek meg.

Nézzünk egy rövid áttekintést a gráfelméleti csoportosításokról. A *Minimális vágás* algoritmusa megpróbál olyan csoportokat kialakítani, ahol a csoportokon belül sok él található, kifelé viszont kevés. Minimális átmérő algoritmusnak hívjuk az algoritmust, ha az minimalizálja a csoportok átmérőjét. (Itt az átmérő a leghosszab legrövidebb utat jelenti a csoporton belüli bármely két pontpár közt. Sok esetben az átlagos távolság jelenti az átmérőt.) Minimális sugár algoritmusnak nevezzük azokat, amelyek minimalizálják a sugarát a csoportoknak. (A csoport sugara a maximális – vagy átlagos – távolság egy megadott középponttól.)
A továbbiakban a minimális sugár algoritmust tárgyaljuk.

4.2 A csoportosító eljárás

Először a csoportosító algoritmust vázolom fel. A részletekről alfejezetekben szólok. Az algoritmus inputja G(V,E) irányított gráf. A cél a gráfpontok csoportosítása. Az algoritmus szétvágja a gráfot V_1, V_2, …, V_k algráfokra. Ugyanezt az algoritmust alkalmazzuk ezután a részgráfokra. A lényege az algoritmusnak a csoport-középpontok megtalálása és a pontok hozzárendelése a legközelebbi középponthoz. Ez a rendezés a középpontok köré építi a csoportokat. A lépései az algoritmusnak:

1. Rendezzük a gráf pontjait, és hívjuk a felső j%-át középpontra jelentkezőknek.

2. Iteráció segítségével kapjuk meg a középpontokat. Elöször a középpontok halmaza üres. Az első középpont az lesz, amelyhez a legtöbb pont van 1 távolságra. Ha a középpont halmaz nem üres, akkor a következő középpont az a pont lesz, amelyiktől a már kiválasztott középpontokhoz hozzáadva a legjobban növeli az 1 távolságra lévő pontok halmazát. Az iteráció megáll, ha a kívánt mennyiségű középpontot megtaláltuk, vagy az új középponttal elért növekedés m% alatt marad.
3. Azt mondjuk, hogy egy jelentkező hasonló egy középponthoz, ha s%-a azoknak a pontoknak amelyek 1 távolságra vannak a középponttól 1 távolságra vannak a jelentkezőktől is. A középpontokkal megegyező jelentkezőket kiszűrjük.
4. Minden pontot hozzárendelünk egy középponthoz. Minden csomópont ahhoz a középponthoz van rendelve, amelyhez a legközelebb van. Azonos távolság esetén véletlenszerű a döntés.
5. A csoportok ponthalmazokból állnak. Ahhoz, hogy hierarchikusan folytatni tudjuk, a középpontokat és a hozzájuk hasonló pontokat nem kell figyelembe venni.

A heurisztika miatt a paraméterek és a konstansok erősen befolyásolják az eredményt, és ezeket a paramétereket hangolni koránt sem egyszerű. A hierarchikus csoportosítás folyamán ez a probléma sokszor fennáll, hiszen az algoritmus minden szint kialakításában szerepet játszik. Viszont egy konzervatív algoritmus leszűkíti a paraméter választást azokra a paraméterekre, amelyek függetlenek a hierarchia szintjétől. Ennek az algoritmusnak az esetében ezek a j (a jelentkezők százaléka), k (az alcsoportok maximális száma), m (min-max ball ratio) és s (hasonlósági arány).

4.3 Sorbarendezés

Az ígéretes jelöltek azok, amelyekből sok más pont elérhető 1 lépés alatt. A legegyszerűbb megoldás, ha a kimenő élek száma szerint rendezzük a pontokat sorba. Egy általánosítása ennek a módszernek az, ha aszerint rendezzük sorba, hogy hány pont érhető el r lépésből. Ezt r-távolság rendezésnek hívjuk (r-distance rank).

Egy másik ígéretes rendezési eljárás a Google által használt Page Rank algoritmus. Részletekbe merülés nélkül megemlíteném, hogy a Page Rank a befokok súlyozott összegével számol, ahol a befoki szomszéd magasabb értékkel nagyobb súllyal számolódik.

Tükrözött Page Rank ugyanezt az algoritmust használja, de a gráfban megfordítjuk az éleket. Ezzel a módszerrel a kifok szerinti rendezés általánosítását kapjuk. Ez a módszer időben költséges, ezért érdemesebb a rendezést megcsinálni, és az előre kiszámolt értékekkel dolgozni mélyebbi szinteken. Érdemes megemlíteni, hogy a Page Rank technológia akkor hasznos, ha a j konstans kicsi, azaz a jelentkezők száma kicsi a többi pont számához viszonyítva.

4.4 Középpontok választása

Mikor középpontokat választunk, az a cél, hogy egy olyan k elemszámú halmazt készítsünk, ahol a legtöbb pont érhető el 1 lépésből a halmazból. Ez egy NP nehéz

optimalizációs feladat. Időmegtakarításhoz érdemesebb mohó algoritmust használni. A hátránya, hogy sub-optimális megoldást kapunk a legtöbb esetben. Másfelöl, a mohó algoritmus legalább k középpontot talál, ami előnyös, mert a mohó algoritmus megállítható k-nál kevesebb lépés után, hogyha az algoritmus többi feltétele teljesült.

Egy általánosítása az algoritmusnak, a sugárnak 1-nél nagyobb értéket választunk. A mohó algoritmust Johnson publikálta a korai 70-es években, hogy középpontokat találjon, amit dominál halmaznak hívunk.

4.5 Hasonló középpontok gyüjtése

A 3. pont összevonó algoritmusa hasonlósági függvényt használ. A hasonlóságot a kapcsolódó szomszédok száma határozza meg. Számos variánst lehetne itt felidézni. nehéz eldönteni tapasztalatok nélkül, hogy melyik eljárás működik a legjobban.

4.6 Hierarchikus csoportosítás

Az 5. pontjában az algoritmusnak a középpontok és a hasonló pontok nem szerepelnek az alsóbb szinteken. Az indok a következő: ha benne hagynánk ezeket a pontokat, akkor a következő szinteknek ugyanezek lennének a középpontjaik. Továbbá a legtöbb pont kevés lépéssel elérhető ezekből a középpontokból. Néhány lépés alatt egy egyedülálló csoport keletkezne. Nézzünk például zenével foglalkozó lapokat. Azt remélnénk, hogy keletkezik például egy klasszikus zenei csoport és egy jazz csoport egy bizonyos mélységen belül. Ellenben a zenei lapok közt vannak olyan általános lapok, amelyek mindkettővel foglalkoznak. Soha nem keletkezne klasszikus és jazz csoport, ha nem hagynánk el ezeket az általános lapokat.

5 Az algoritmus vizsgálata

A csoportosító algoritmus vizsgálatánál két kérdés merült fel:

- melyik csoportosító módszer
- melyik kulcsszó-kereső eljárás

a legjobb különböző adatbázisokon a fontos lapok kikeresésében. Van-e győztes algoritmus, vagy algoritmusok kombinációjára van szükség az emberi és roboti intelligencia fúziójához.
Két alapvető módszert próbáltam ki:

- **FAQ és válaszok:**
  Egy FAQ (Frequently Asked Questions, gyakran ismételt kérdések) szavait és azok hipernimáit úgy választottam ki, hogy eldobtam a gyakori szavakat, mint a 'the', 'a', stb. Ezután hasonló szavakkal rendelkező oldalakat kerestem. Ezek után tanulmányoztam, hogy a megtalált oldalak szavai mennyire jól illeszkednek a FAQ válaszainak a szavaihoz.
    - Az értékelést hipernimával és anélkül
    - különböző adatbázisokon

  Ez egy 'brute-force' tanulmány volt, hogy felállítsak egy 'szakértő-nem szakértő' szótárt (expert-non expert, EnE szótár). Az eredmény az lett, hogy a hipernimák önmagukban segíthetnek. A legtöbb esetben ez a módszer használható, mert mind a FAQ, mind a ráadott válaszok ugyanolyan stílusban íródtak. A hipernimák halmazát a Word Associationt általános esetekre találták ki, és nincsenek benne pontos szinonímák új technikai és orvosi szavakra. Másféle hozzáállás szükséges, hogy egy 'szakértő-nem szakértő' szótárt összeállítsunk. Az új algoritmusok az Internet dokumentumait használják, amik használhatók kézi készítésű hipernima gyüjtemények felváltására és frissek. Néhány eljárást érdemes megvizsgálni.
- **Csoportosítás és kulcsszavak:**
  Különböző csoportosító és kulcsszókereső módszert használtam arra, hogy megtaláljam azokat a szavakat, amelyek jól reprezentálják a csoportokat. Más szóval különböző csoportosító eljárásokat értékeltem, hogy kiderítsem, téma-függőek-e. Azt találtam, hogy a Page Rank nem alkalmas ilyenfajta csoportosításra, a direkt módszerek jobb eredményt produkáltak.

A fő konklúciója a munkának, hogy az összes korszerű algoritmusnak megvan az igérete, de egyik sem elegendő önmagában, és a felhasználó-specifikus egyesítésük cél-orientált kombinációját igényli ezen és ezekhez hasonló módszereknek. A megerősítéses tanulás igéretesen használható lehet ebben a problémakörben.

5.1 Vizsgált adatbázisok

1) **American Heart Association**
   - Rövid név: AHA
   - Cím: http://www.americanheart.org
   - Ellátva jó kulcsszavakkal és kifejezésekkel
2) **Center for devices and radiological health**
   - Rövid név: General
   - Cím: http://www.fda.gov/cdrh/

- Közepesen ellátva kulcsszavakkal

3) **Collaborative Hypertext of Radiology**
   - Rövid név: Radiology
   - Cím: http://chorus.rad.mcw.edu
   - Közepesen ellátva kulcsszavakkal

5.2 Alkalmazott csoportosító eljárások

A csoportosító eljárások a középpontok kiválasztásában különböznek. Minden algoritmus más-más sorrendbe állítja a pontokat, így mások lesznek a középpontok.

1) **Greedy outdeg**
   - A legtöbb kifelé mutató éllel rendelkező nódus lesz kiválasztva.
   - Ez a 'következő nódus'-a a gráfnak.
   - Ezt eltávolítva a gráfból kapjuk meg a 'következő gráf'-ot.
   - Minden nódusra újra kiszámítjuk a kimenő éleket a 'következő gráf'-ban.
2) **outdeg**
   - A nódusok a kimenő élek szerint rendezettek.
   - Nem módosítjuk a gráfot.
3) **indeg**
   - A nódusok a bejövő élek szerint rendezettek.
4) **page rank**
   - A nódusok az eredeti Page Rank algoritmus szerint rendezettek. Az algoritmus nem veszi figyelembe az irányítást az éleken.

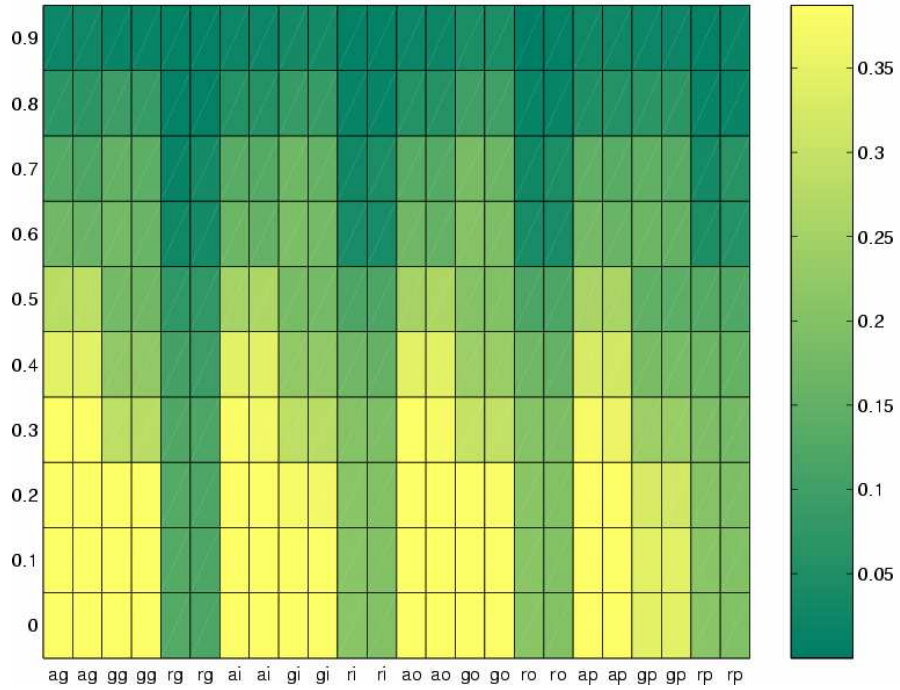
5.3 Kulcsszavak, kulcs-kifejezések generálása

A kérdés az, hogy különböző kulcsszó-kereső algoritmusok és a különböző csoportosító eljárások milyen jó segítik a releváns dokumentumok megtalálását. A következő módszereket használtuk:

1. **Kezdőlap kategóriák:** A kezdőlapja minden adatbázisnak tartalmaznak kategóriákat. Ebben az esetben ezek a kategóriákat és a lap szövegét használtuk kulcsszavakként az adatbázishoz.
2. **Kölcsönös Információ (Mutual Information, MI) a csoportok szavai alapján:** Az algoritmus lényege: A csoportok dokumentumokat tartalmaznak, a dokumentumok szavakat. Az MI statisztikus eljárás kiszámolja, hogy egy szó mennyire jól reprezentál egy csoportot a többi ellenében. Minden szóra kiszámolja az értéket. A legjobb 100 szót kiválasztjuk, mint az adatbázis kulcsszavait.
3. **A 'keyword' META-tag szavai:** Az AHA adatbázis majdnem minden dokumentuma tartalmaz XML-szerű META tagot, amely megadja a kulcsszavait az oldalnak. Például: <meta name="keywords" content="cholesterol,stroke,heart disease,risk factors"> A General és Radiology adatbázisok néhány lapja is tartalmaz ilyen kulcsszavakat. Ezeket a szavakat használjuk, mint az adatbázis kulcsszavait.

5.4 Eredmények, áttekintés

5.4.1 A „FAQ – válasz” módszer

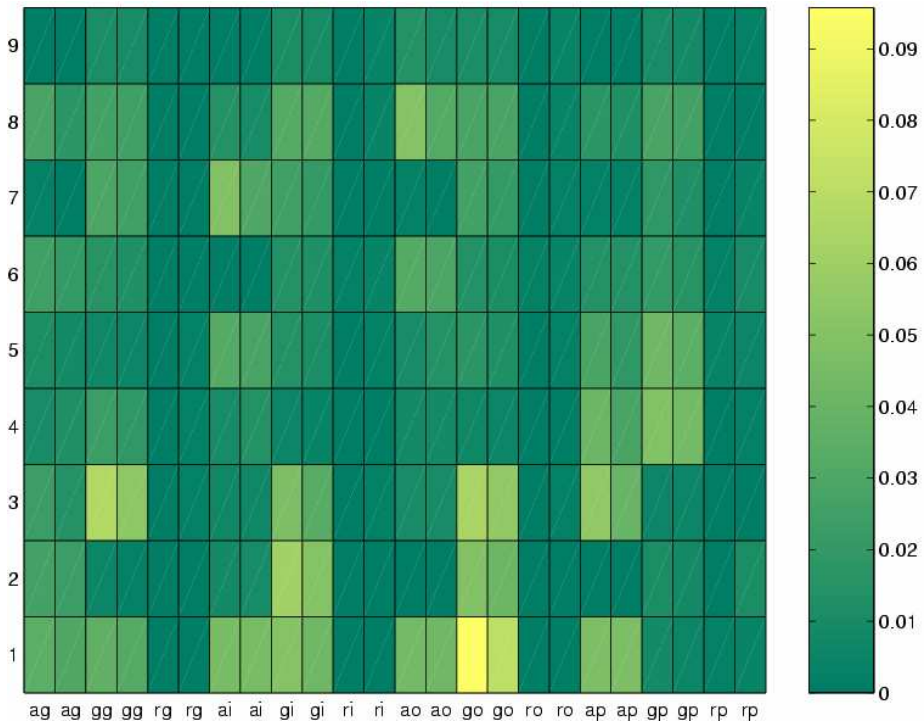


2. ábra A FAQ - válasz eljárás értékelése

Adatbázis: mind a három adatbázis. Az első betű a vizszintes tengelyen: a=AHA, g=Gneral, r=Radiology
Az eredmények átlagoltak csoportra és FAQ-ra.
Függőleges tengely: küszöbértékek.
Csoportosító módszer: Minden csoportosító módszer: A második betű a vízszintes tengelyen: g=greedy_outdeg, i=indeg, o=outdeg, p=PageRank
A FAQ szám első megjelenéséhez tartozó oszlop: kérdés hipernimák nélkül
A FAQ szám második megjelenéséhez tartozó oszlop: kérdés hipernimákkal
A szinek a négyzetes különbségét ábrázolják csoport értékének kérdésekre és a csoport értékének válaszokra.
Az értékek számolásához szó frekvencia előfordulást használtunk.
A szín értékek a jobb oldalioszlopban találhatók.

A sötétebb szinek a jobb találatok. Látható, hogy a hipernimák használata alig okozott változást. Három megjegyzés:

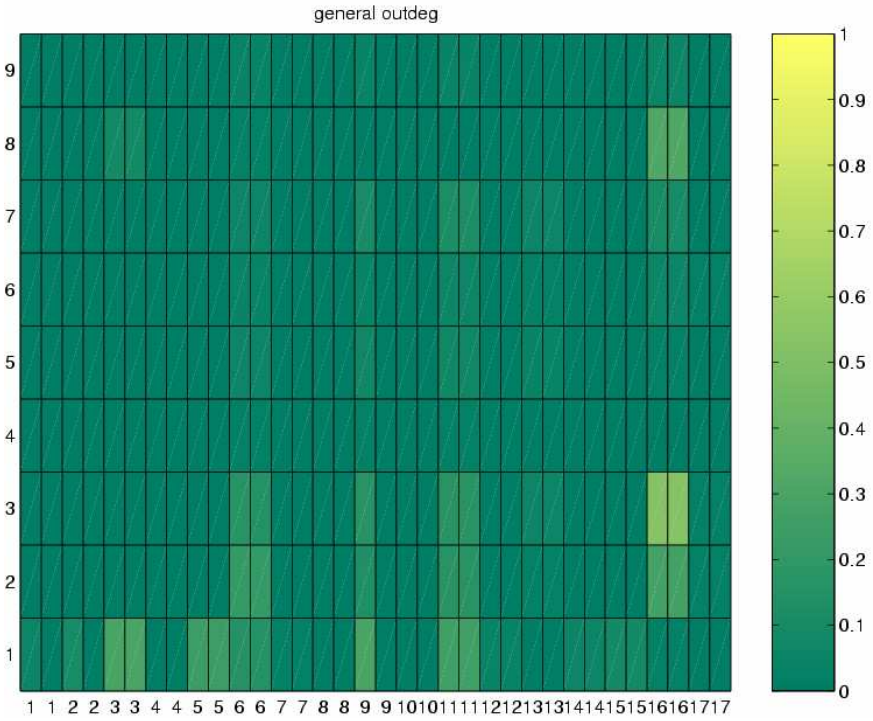
1. A különbségek függenek a csoportosító módszertől
2. Más ábrákon látszik, hogy a FAQ-válasz módszer nem hatékony csoport reprezentációjára.
3. A 2. pontot elronthatja
    a. A FAQ-k alacsony száma
    b. A Word Associations hipernimái, amelyek nem tükrözték vissza jól a releváns szót.



3. ábra A FAQ - válasz eljárás értékelése

Adatbázis: mind a három adatbázis. Az első betű a vizszintes tengelyen: a=AHA, g=Gneral, r=Radiology Az eredmények átlagoltak FAQ-ra.
Csoportosító módszer: Minden csoportosító módszer: A második betű a vízszintes tengelyen: g=greedy_outdeg, i=indeg, o=outdeg, p=PageRank
A FAQ szám első megjelenéséhez tartozó oszlop: kérdés hipernimák nélkül.
A FAQ szám második megjelenéséhez tartozó oszlop: kérdés hipernimákkal.
Függőleges tengely: csoport szám (kisebb szám, kisebb csportméret)
Küszöbérték: 0.7
A szinek a négyzetes különbségét ábrázolják csoport értékének kérdésekre és a csoport értékének válaszokra.
Az értékek számolásához szó frekvencia előfordulást használtunk.
A szín értékek a jobb oldalioszlopban találhatók.

Itt is a sötétebb szin a jobb érték. Látszik, hogy a hyerniym-ák kevés helyen tudtak segíteni. A 2.ábra megjegyzése itt is érvényes. Megvizsgáljuk a hatást a General adatbázison outdeg módszer mellett a következő ábrán.

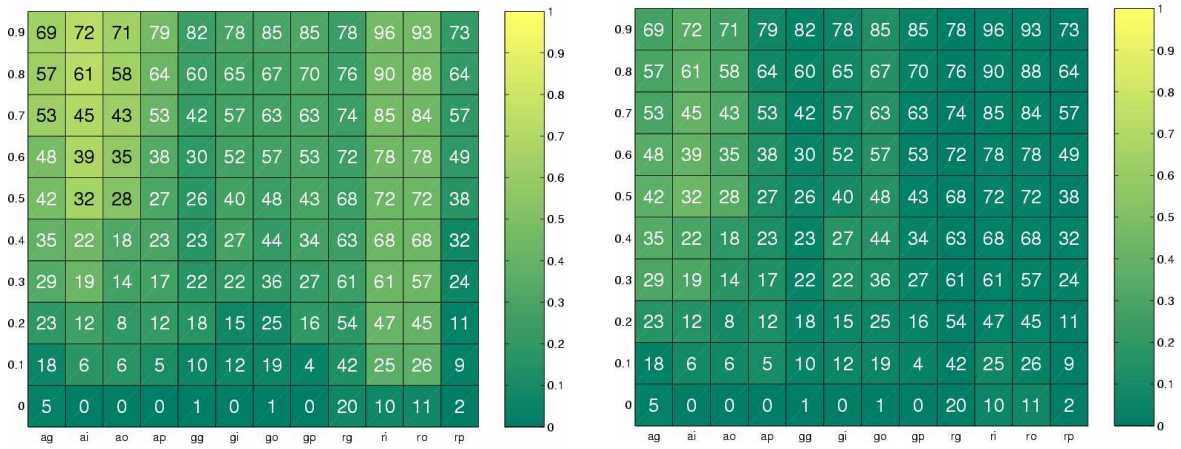


4. ábra A FAQ - válasz eljárás értékelése

Adatbázis: General (Center for devices and radiological health)
Csoportosító eljárás: outdeg
Vízszintes tengely: A FAQ sorszáma
A FAQ szám első megjelenéséhez tartozó oszlop: kérdés hipernimák nélkül.
A FAQ szám második megjelenéséhez tartozó oszlop: kérdés hipernimákkal.
Függőleges tengely: csoport szám (kisebb szám, kisebb csportméret)
Küszöbérték: 0.7
A szinek a négyzetes különbségét ábrázolják csoport értékének kérdésekre és a csoport értékének válaszokra.
Az értékek számolásához szó frekvencia előfordulást használtunk.
A szín értékek a jobb oldalioszlopban találhatók.

A 4. ábra szerint a javulás, amit megfigyeltünk a 3. ábrán, főként a 9. FAQ kérdésnek volt köszönhető. A javítás, amit a hipernimákkal elérhetünk, behatárolt. A megjegyzés a hipernimák minőségére megalapozott. Különösen, a kevés FAQ mutatja, a FAQ reprezentál*hat*ja a csoportokat. Ez a brute-force eljárás önmagában azonban nem elégséges, hogy reprezentálja a csoportokat.

5.4.2 Kulcsszavak és kulkcskifejezések módszer



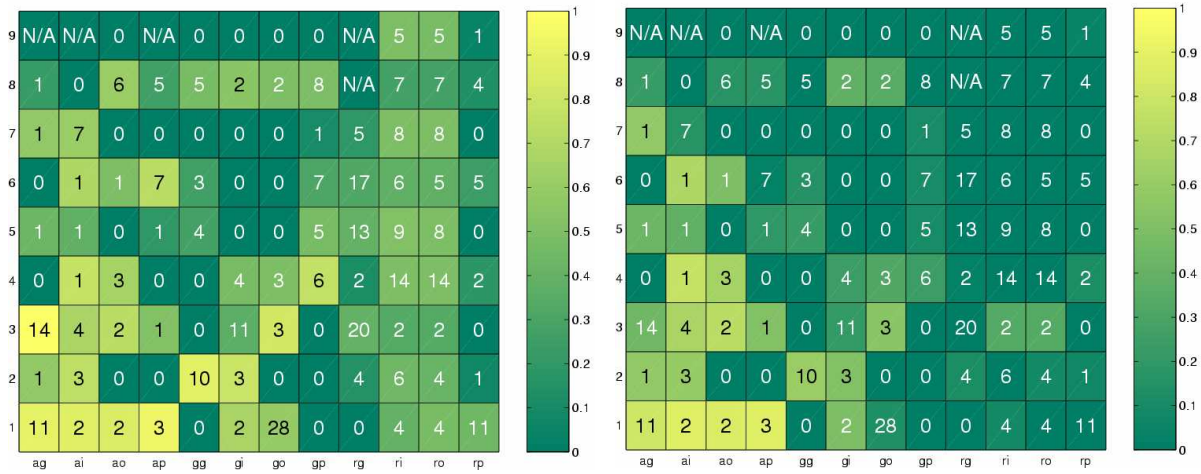
5. ábra Kulcsszavak és kulcskifejezések Kölcsönös Információ segítségével:módszerek és küszöbök

Bal oldal: VAGY kapcsolat a kulcsszavak közt
Jobb oldal: ÉS kapcsolat a kulcsszavak közt

Adatbázis: mind a három adatbázis. Az első betű a vizszintes tengelyen: a=AHA, g=Gneral, r=Radiology Az eredmények átlagoltak FAQ-ra.
Csoportosító módszer: Minden csoportosító módszer: A második betű a vízszintes tengelyen: g=greedy_outdeg, i=indeg, o=outdeg, p=PageRank
Függőleges tengely: küszöb-értékek
Számok a területeken belül: A kinyert kulcsszavak a megfelelő küszöb mellett (vízszintesen), a megfelelő adatbázisban (függőleges első betű), a megfelelő módszerrel (függőleges második betű)
A szinek átlagolt csoport-értékeket mutatnak.

Az 5. ábra megmutatja, hogy a Kölcsönös Információval (Mutual Info) jó esélyünk van olyan számokat választani, amik reprezentálják a csoportokat. A feladat, hogy megfelelő küszöböt találjunk. A Kölcsönös Információ alapú kulcsszóválasztás egy kicsit jobb, mint az adatbázis kezdőoldala alapján választott kulcsszavak. Ez azt igéri, hogy ezek az adatbázisok jól megalapozottak, és az igazi feladat a még nem kategorizált csoportokat érinti. Ebben a vonatkozásban, a kulcsszavak amiket kölcsönös információval találtunk, jobban jellemzik a csoportokat, mint a adatbázis tervezői által kitaláltak.
Vonatkozó megjegyzések:

1. A PageRank (4., 8. és 12. oszlop) kevésbé jellegzetes, mint a többi oszlop.
2. A Radiology adatbázisban az ÉS kapcsolat jelentés nélküli minden tipusú csoportosításra. Ez összecsneg azzal a megfigyeléssel, hogy a VAGY kapcsolat a Radiology adatbázison felette áll az összes többi adatbázisnak.



6. ábra Kulcsszavak és kulcskifejezések kölcsönös információ segítségével: önálló csoportok

Bal oldal: VAGY kapcsolat a kulcsszavak közt
Jobb oldal: ÉS kapcsolat a kulcsszavak közt

Adatbázis: mind a három adatbázis. Az első betű a vizszintes tengelyen: a=AHA, g=Gneral, r=Radiology Az eredmények átlagoltak FAQ-ra.
Csoportosító módszer: Minden csoportosító módszer: A második betű a vízszintes tengelyen: g=greedy_outdeg, i=indeg, o=outdeg, p=PageRank
Küszöbérték: 0.3
Függőleges tengely: Csoport száma. A kisebb szám kisebb méretű csoportot jelöl. (Az N/A azt jelöli, hogy némelyik csoportosító eljárás kevesebb, mint 9 csoportot készít)
A számok a területen belül: A kinyert kulcsszavak száma a csoporton belül (vízszintesen), a megfelelő adatbázisban (függőlegesen az első betű), a megfelelő eljárással (függőlegesen a második betű).
A szinek átlagolt csoport-értékeket mutatnak.

A 6.ábra megmutatja, hogy az 5. ábra releváns. Emlékezzünk, hogy a FAQ-válasz módszerben a javítások csak egy csoportnál volt megfigyelhető. A kölcsönös információval kinyert kulcsszavak alkalmasak arra, hogy reprezentáljanak és elhatároljanak több csoportot, mind a VAGY, mind az ÉS kapcsolat segítségével. Azaz, vannak eljárások, amik indexeket állítanak elő, ami az első lépés, hogy a robot-intelligencia érthető legyen az ember számára.

6 Függelék

6.1 A Top-down csoportosító algoritmus pszeudo-kódja

A pszeudo-kódnak 3 fő része van:

1. Nódusértékek számolása (cluster-koefficiens) a következők alapján:
    a. PageRank (mind kimenő, mind bejövő linkek)
    b. kimenő linkek
    c. bejövő linkek
2. Megkeresi a középpont jelöltjeit, aztán a középpontjait a csoportnak
3. Megépíti a csoportot

```
main
  INPUT
    adj ← the adjacency matrix of the graph to be clustered
    N ← number of nodes
    valueMode ← the method modification for node value computation
    candidateRatio ← the ratio of cluster center candidate node
    minMaxSizeRatio ← the maximal relative di®erence in cluster size
            estimation while selecting cluster centers
    maxSimilarityRatio ← the maximal relative size of the common part
            of two cluster center neighborhoods
    maxDepth ← the maximum step number from cluster centers
  endINPUT

  nodeValues ← calculateNodeValues(adj, valueMode)
  centerCandidates ← the candidateRatio*N highest valued node
  centers ← selectCenters(adj, centerCandidates, minMaxSizeRatio,
                          maxSimilarityRatio)

  clusters ← buildCluster(adj, centers, maxDepth)
  if there are not clustered nodes
    c ← the highest valued not clustered node
    centers ← centers ∪ c
    clusters(c) ← the remaining not clustered nodes
  endif
  return centers, clusters
end main
```

```
nodeValues = calculateNodeValues(adj, valueMode)
  if valueMode = outdeg or indeg or pagerank
    for each node n
      nodeValues(n) ← (number of outgoing or incoming links)
                       or pagerank value of n
    endfor
  else if valueMode = greedyout
    while there is node in adj
      n ← the node with highest outgoing link number in adj
      nodeValues(n) = number of outgoing links of n
      adj ← leave n from adj with it's links
    endwhile
  endif
end calculateNodeValues
```

```
centers = selectCenters(adj, centerCandidates, minMaxSizeRatio,
           maxSimilarityRatio)
  for each centerCandidates cc
    clusters(cc) ← the nodes which are accessible from cc
                   in adj in 1 step and are not centerCandidates
  endfor
  maxClusterSize ← max_cc|clusters(cc)|
  for each centerCandidates cc
    if |clusters(cc)| < minMaxSizeRatio*maxClusterSize
      centerCandidates ← remove cc from centerCandidates
    endif
  endfor
  do
    wasJoining ← false
    minSize ← infinity
    for each centerCandidates cc1 and cc2,cc1 ≠ cc2
      commonSize ← |clusters(cc1 ) ∩ clusters(cc2 )|
      actMinSize ← min(|clusters(cc1 )|,|clusters(cc2 )|)
      if actMinSize<minSize or commonSize>maxSimilarityRatio*actMinSize
        minSize ← actMinSize
        join1 ← cc1
        join2 ← cc2
      endif
    endfor
    if minSize < infinity
      wasJoining ← true
      clusters(join1 ) ← clusters(join1 ) ∪ clusters(join2 )
      join1 ← join1 ∪ join2 (join1 's cluster will have more centers:
                 the centers denoted by join1 plus the centers
                 denoted by join2)
      centerCandidates ← remove join2 from centerCandidates
    endif
  while wasJoining
  for each centerCandidates cc
    if |clusters(cc)| < minMaxSizeRatio*maxClusterSize
      centerCandidates ← remove cc from centerCandidates
    endif
  endfor
  centers ← centerCandidates
end selectCenters
```

```
clusters = buildCluster(adj, centers, maxDepth)
  for each center c
    clusters(c) ← the nodes which are accessible from c in
                  adj and are not centers
  endfor
  for each node in adj n
    if n is in more clusters
      if there is a unique center from where the distance is smallest
        remove n from other clusters
      else
        c ← random cluster
        remove n from the clusters but c
      endif
    endif
  endfor
end buildCluster
```

6.2 A kérdés-értékelés pszeudokódja

A csoportok értéke (a releváns dokumentumok száma a csoport méretéhez viszonyítva) nem változott jelentősen, ha kiterjesztettük a kérdéseket a szavak hipernimáival. Négy területet különböztetünk meg a θ küszöbérték függvényében:

Region 1: $\theta \in (0...0.4)$: Túl sok dokumentum maradt.
Region 2: $\theta \in (0.4...0.6)$: A jó dokumentumok száma gyorsan csökken.
Region 3: $\theta \in (0.6...0.8)$: A jó dokumentumok száma lassú függvénye θ-nak. A hipernimák segítettek néhány esetben.
Region 4: $\theta \in (0.8...1)$: Túl kevés dokumentum maradt.

```
main
  INPUT
    θ ∈ 0; 0:1; 0:2; …; 0:9 threshold parameter
    CLUSTER DOC(c) ← documents in cluster c
    questions
    answers
  endINPUT

  remove stopwords from questions and answers
  remove all words from questions wich are not in the expert dictionary
  TF_QWOA ← term frequency vectors of questions (without associations)
  TF_QWA ← term frequency vectors of extended questions with
            associations from WordNet
  TF_AWOA ← term frequency vectors of answers (without associations)
  TF_DOC ← term frequency vectors of documents
  for each document d and for each question q
    DOC_VALUE_QWOA(d,q) ← scalar product of TF_QWOA(q) and TF_DOC(d)
    DOC_VALUE_QWA(d,q) ← scalar product of TF_QWA(q) and TF_DOC(d)
  endfor
  for each document d and for each answer a
    DOC_VALUE_AWOA(d,a) ← scalar product of TF_AWOA(a) and TF_DOC(d)
  endfor
  for each question q
    MAX_DOC_VALUE_QWOA(q) ← max_d DOC_VALUE_QWOA(d,q)
    MAX_DOC_VALUE_QWA(q) ← max_d DOC_VALUE_QWA(d,q)
    GOOD_DOC_QWOA ← empty set
    for all DOC VALUE QWOA(d,q) ≥ θMAX_DOC_VALUE_QWOA(q)
      GOOD_DOC_QWOA(q) ← GOOD_DOC_QWOA ∪ d
    endfor
    GOOD_DOC_QWA ← empty set
    for all DOC_VALUE_QWA(d,q) ≥ θMAX_DOC_VALUE_QWA(q)
      GOOD_DOC_QWA(q) ← GOOD_DOC_QWA ∪ d
    endfor
  endfor
  for each answer a
    MAX_DOC_VALUE_AWOA(a) ← max_d DOC_VALUE_AWOA(d,a)
    GOOD_DOC_AWOA ← empty set
    for all DOC_VALUE AWOA(d,a) ≥ θMAX_DOC_VALUE_AWOA
      GOOD_DOC_AWOA(a) ← GOOD_DOC_AWOA ∪ d
    endfor
  endfor
```

$$\text{CLUSTER\_VALUE\_QWOA}(c,q) \leftarrow \frac{\left|CLUSTER\_DOC(c) \cap GOOD\_DOC\_QWOA(q)\right|}{\left|CLUSTER\_DOC(c)\right|}$$

$$\text{CLUSTER\_VALUE\_QWA}(c,q) \leftarrow \frac{\left|CLUSTER\_DOC(c) \cap GOOD\_DOC\_QWA(q)\right|}{\left|CLUSTER\_DOC(c)\right|}$$

$$\text{CLUSTER\_VALUE\_AWOA}(c,a) \leftarrow \frac{\left|CLUSTER\_DOC(c) \cap GOOD\_DOC\_AWOA(q)\right|}{\left|CLUSTER\_DOC(c)\right|}$$

```
  return CLUSTER_VALUE_QWOA, CLUSTER_VALUE_QWA, CLUSTER_VALUE_AWOA
end main
```

6.3 A kulcsszó-értékelés pszeudo-kódja

Az eredmények jó humán felhasználóknak, alacsony küszöbérték mellet (0.2 vagy 0.3). A magas küszöbérték azt jelenti, hogy sok dokumentumot elzárunk a felhasználó elől. A legtöbb esetben a kölcsönös információ alapú kiértékelés jobb volt, mint a kezdőoldal alapú, azaz nagyobb százalékban azonosított kulcsszó alapján minden csoportban. Radiology adatbázisnál indeg és outdeg módszernél $\theta = 0.3$ küszöbnél a kulcsszavak majdnem ugyanazok voltak. Az első két kulcsszó gyüjtés tipusnál a PageRank módszer nem hozott olyan jó eredményt, mint a másik három. A következő kulcsszó gyüjtési tipusnál a csoport-azonosítás jobb volt VAGY kapcsolatban. Ez a lehetséges kulcsszavak nagy száma miatt volt. Az ÉS kapcsolat nem produkált használható csoport azonosítást.

```
main
  INPUT
    θ ∈ 0, 0.1, 0.2, …, 0.9 threshold parameter
    CLUSTER_DOC(c) ← documents in cluster c
    KEYPHRASES ← phrases
  endINPUT
  for each cluster c
    CLUSTER_KEYPHRASES(c) ← empty set
  endfor
  for each KEYPHRASE k
    GOOD_DOCUMENTS(k) ← set of documents containing k
    CLUSTER_VALUE(c,k) ← |GOOD_DOCUMENTS(k) ∩ CLUSTER_DOC(c)| / |CLUSTER_DOC(c)|
    MAX_CLUSTER_VALUE ← max_c CLUSTER_VALUE(c,k)
    GOOD_CLUSTERS ← empty set
    for all CLUSTER_VALUE(c,k) ≥ θMAX_CLUSTER_VALUE
      GOOD_CLUSTERS ← GOOD_CLUSTERS ∪ c
    endfor
    if |GOOD CLUSTERS| = 1
      c ← the only element of GOOD_CLUSTERS
      CLUSTER_KEYPHRASES(c) ← CLUSTER_KEYPHRASES(c) ∪ k
    else
      KEYPHRASES ← remove k from KEYPHRASES

    endif
  endfor
  for each cluster c
    for all documents d ∈ CLUSTER_DOCUMENTS(c)
      for all keyphrase k ∈ CLUSTER_KEYPHRASES(c) or d contains k

        GOOD_DOCUMENTS(k) ← GOOD_DOCUMENTS(k) ∪ d
      endfor
    endfor
    CLUSTER_VALUES_AND(c) ← |∩_k GOOD_DOCUMENTS(k)| / |CLUSTER_DOC(c)|
    CLUSTER_VALUES_OR(c) ← |∪_k GOOD_DOCUMENTS(k)| / |CLUSTER_DOC(c)|
  endfor
  return CLUSTER_KEYPHRASES, CLUSTER_VALUES_AND, CLUSTER_VALUES_OR
end main
```

6.4 A kölcsönös információ alapú kulcsszó keresés pszeudo kódja

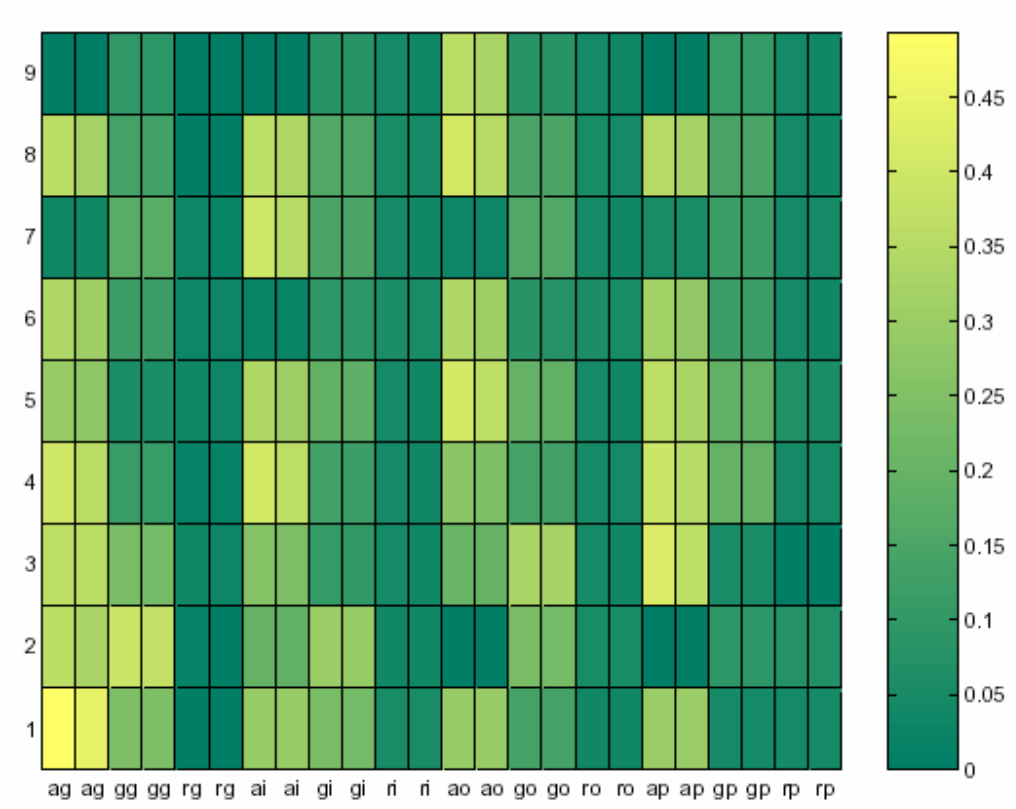
```
main
  INPUT
    C =& set of clusters
    D =& set of documents
    W =& set of the words from which the algorithm
              selects the best 100 one
    T(d): D → C the clustering, i.e. decides
                 which document belongs to which cluster
    H(d,w): D × W → {0,1} returns 1 if d contains w;otherwise returns 0
    Pr(prop) gives back the probability of the given proposition
  endINPUT

  for each word w ∈ W
    MI(w) ← Σ_{c∈C} Pr(T(d) = c, H(d,w) = 0) · log(Pr(T(d) = c | H(d,w) = 0)) +
    Σ_{c∈C} Pr(T(d) = c, H(d,w) = 1) · log(Pr(T(d) = c | H(d,w) = 1))
  endfor
  GOOD_WORDS ← set of 100 w ∈ W words with highest MI(w) value
  return GOOD_WORDS
end main
```

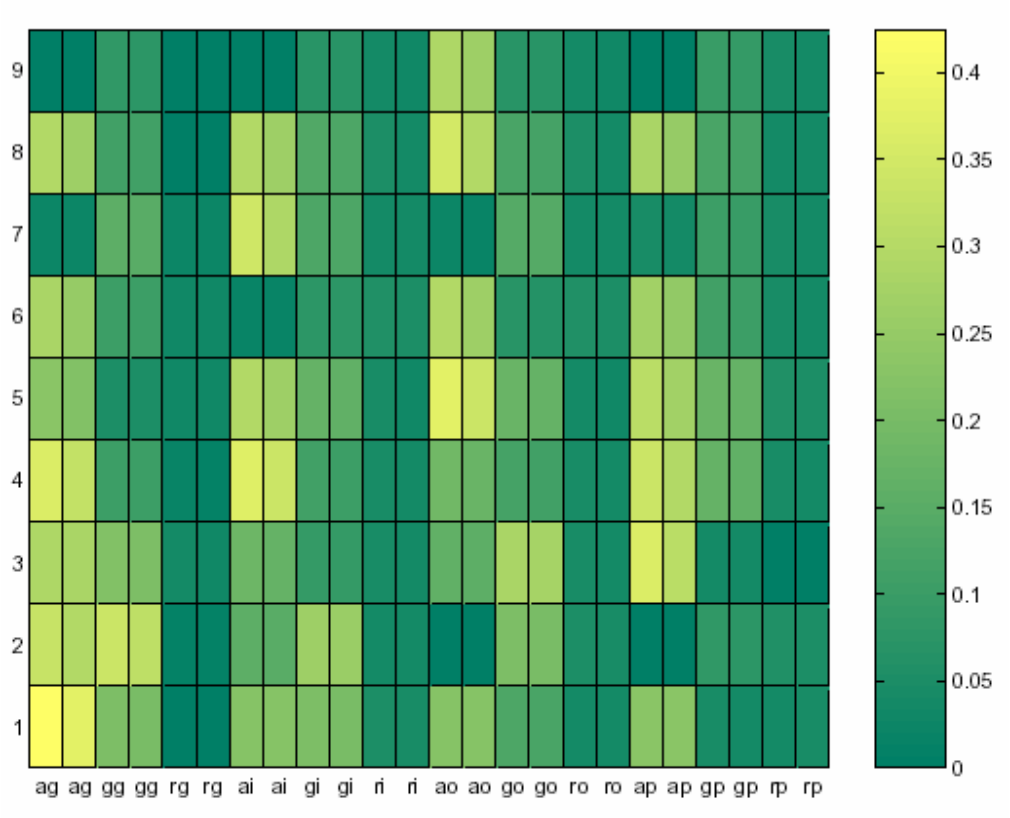
6.5 Ábrák

Megjegyzések az ábrákhoz:

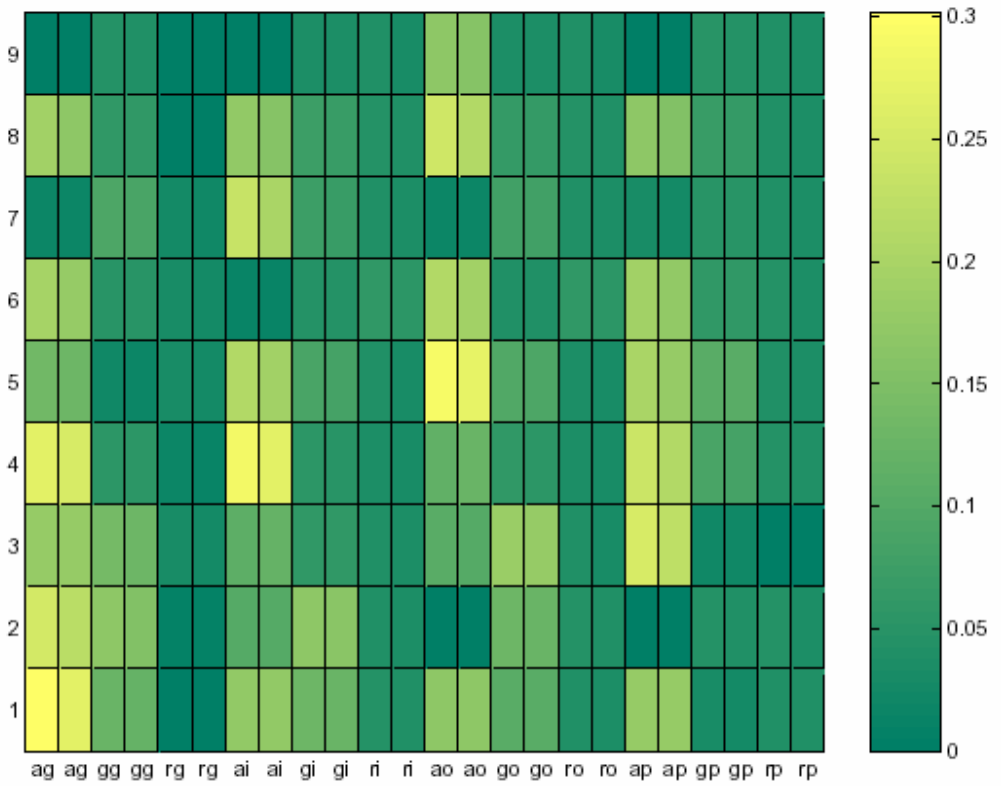
- **a:** AHA adatbázis
- **g az első helyen:** General adatbázis
- **r:** Radiology adatbázis
- **g a második helyen:** greedy outdeg
- **o:** outdeg
- **i:** indeg
- **p:** Page Rank
- **1 1 1 2 2 2 … 17 17 17:** Az 1., 2., …, 17. kérdés asszociációk nélkül (QWOA), asszociációkkal (QWA), és a megfelelő válasz (AWOA)
- **0…9 az Y tengelyen:** Csoport index. A nagyobb index, több dokumentet jelent.
- **0.0…0.9 az Y tengelyen:** Küszöbértékek
- **Számok a cellákban:** A csoporthoz hozzárendelt kulcsszavak
- **N/A a cellákban:** Abban az adatbázisban a cella aktuális eljárása kevesebb, mint 9 csoportot képezett, azaz a cella nem reprezentál egy csoportot sem.



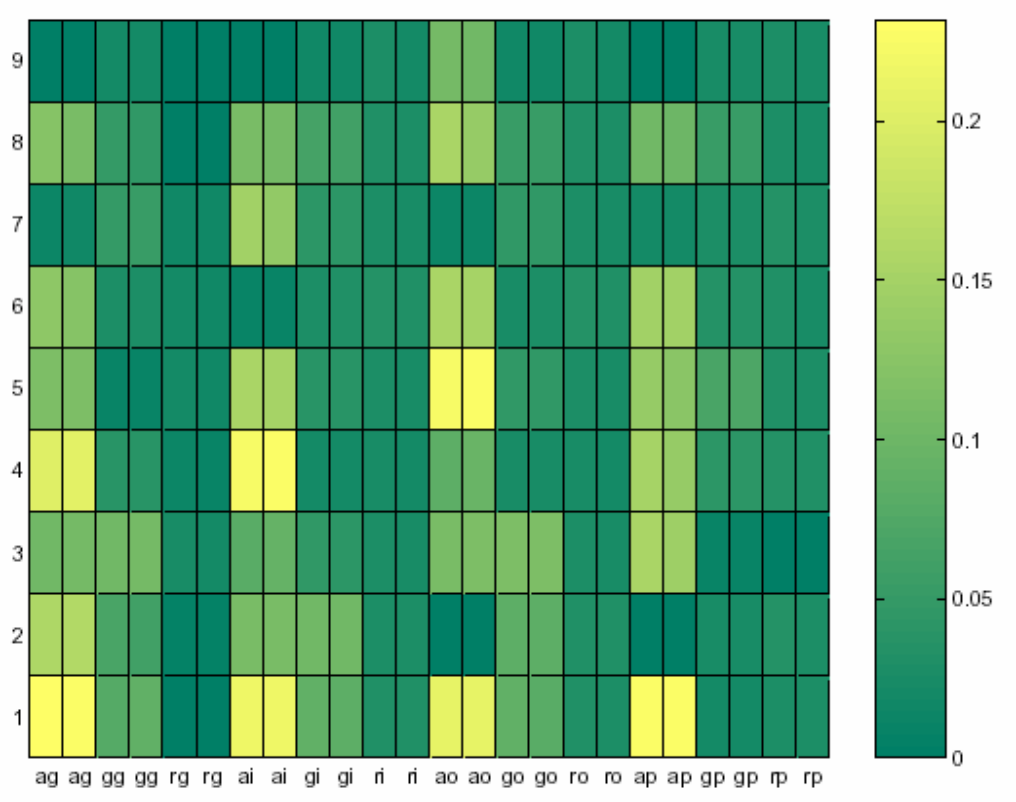
7. ábra Átlagolt QA különbség értékek 0.1-es küszöbnél



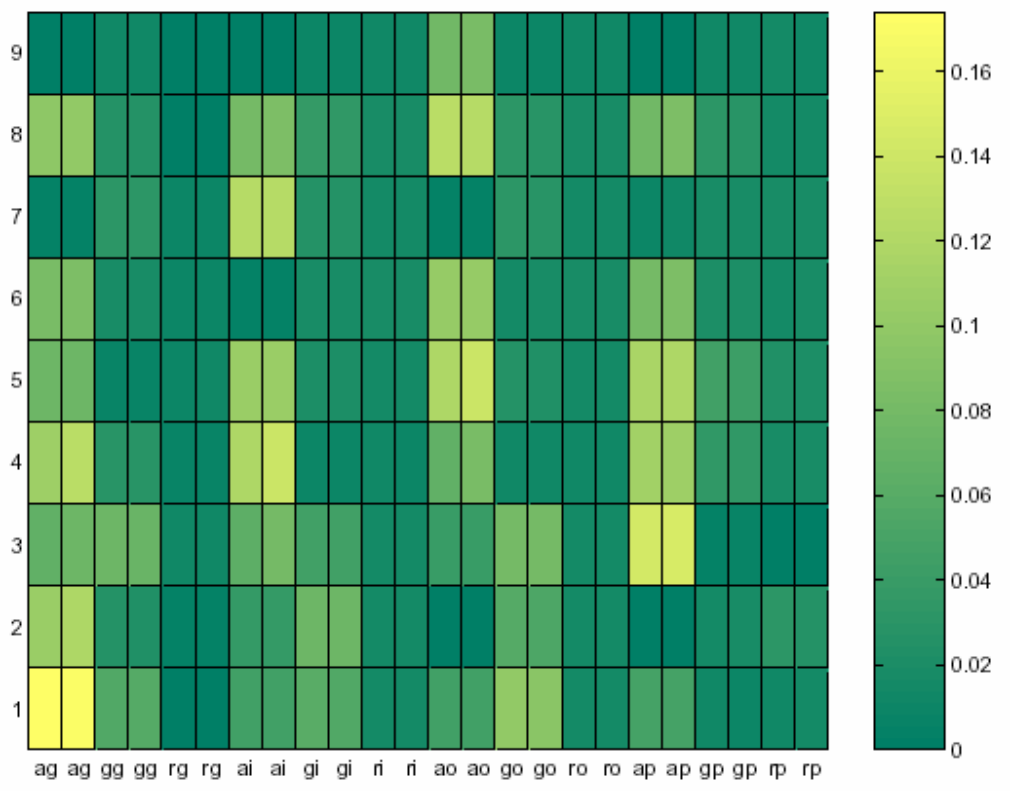
8. ábra Átlagolt QA különbség értékek 0.2-es küszöbnél



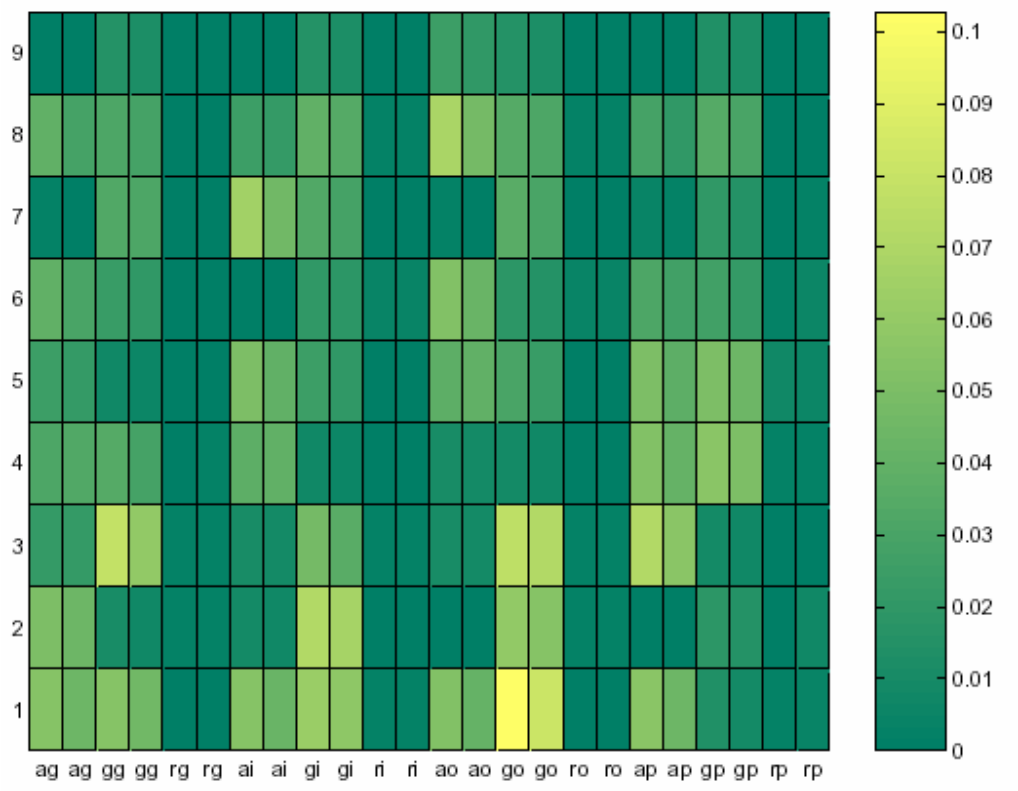
9. ábra Átlagolt QA különbség értékek 0.3-as küszöbnél



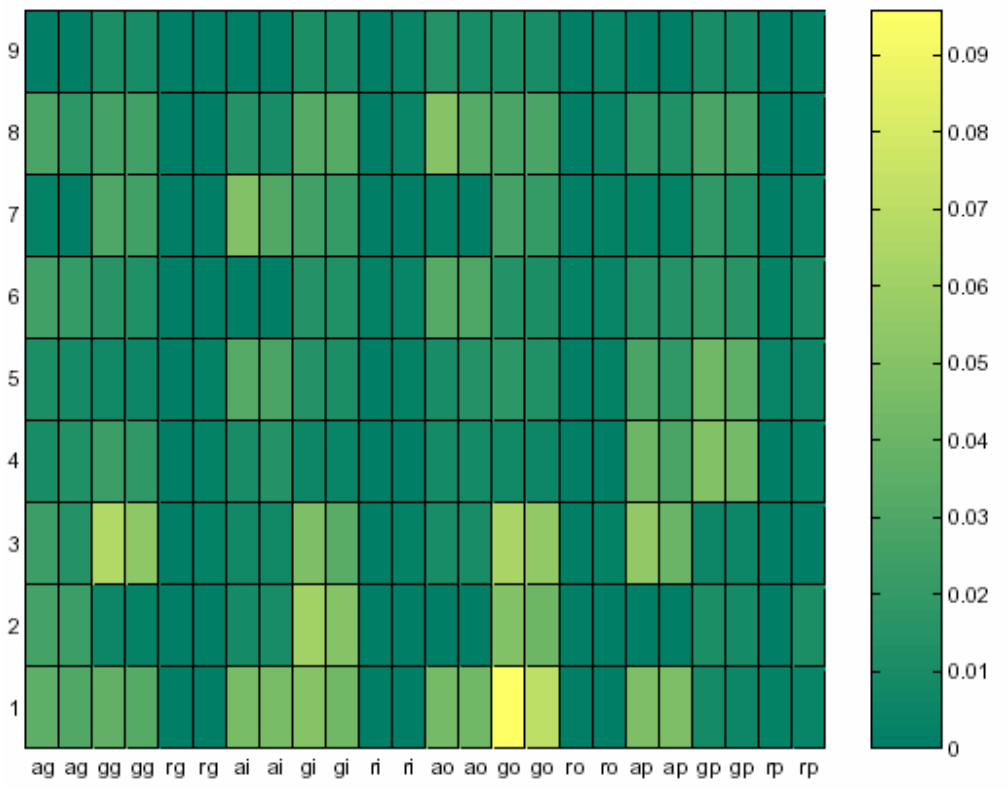
10. ábra Átlagolt QA különbség értékek 0.4-es küszöbnél



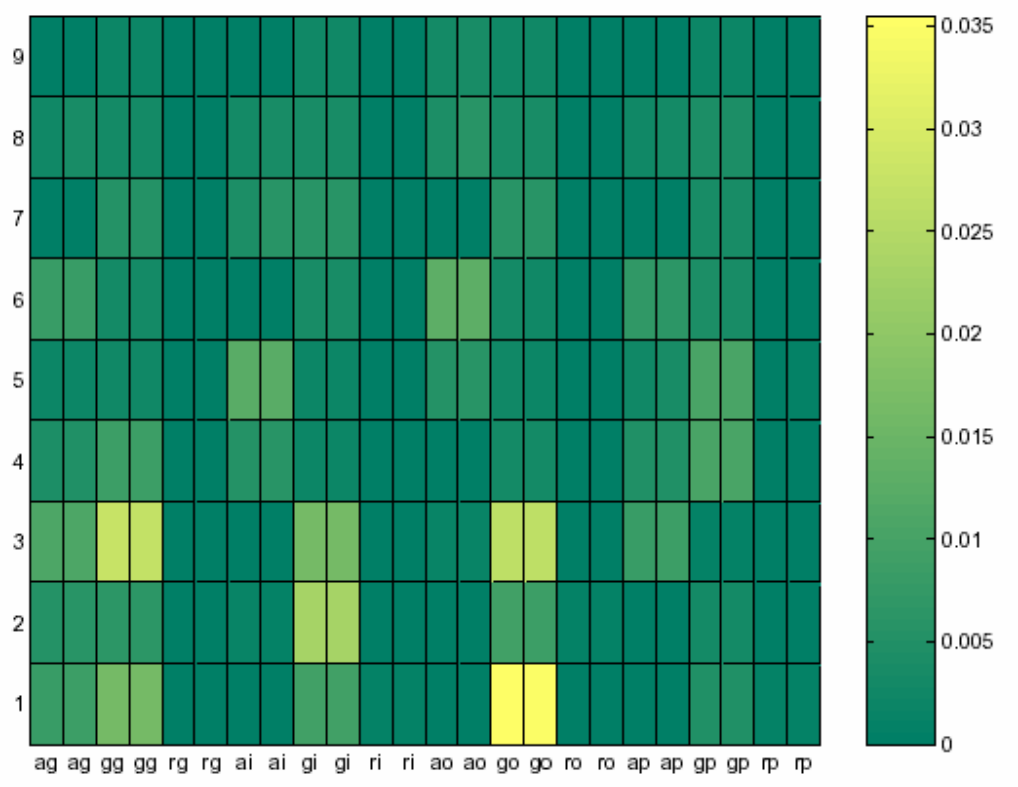
11. ábra Átlagolt QA különbség értékek 0.5-ös küszöbnél



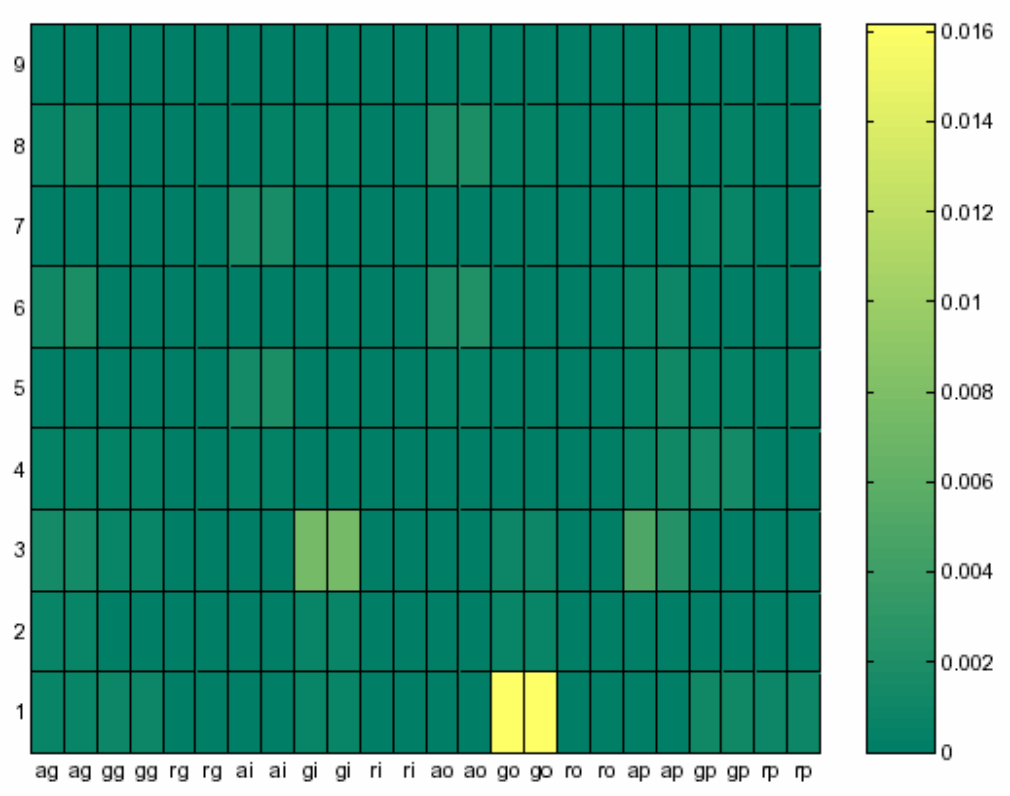
12. ábra Átlagolt QA különbség értékek 0.6-os küszöbnél



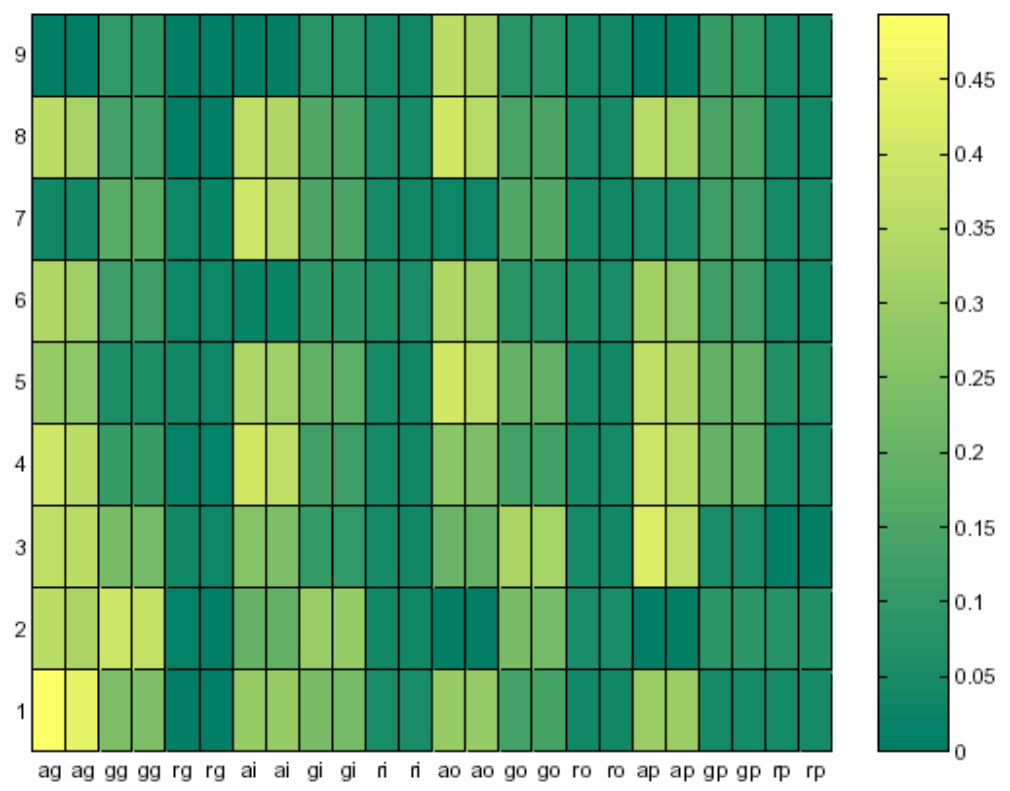
13. ábra Átlagolt QA különbség értékek 0.7-es küszöbnél



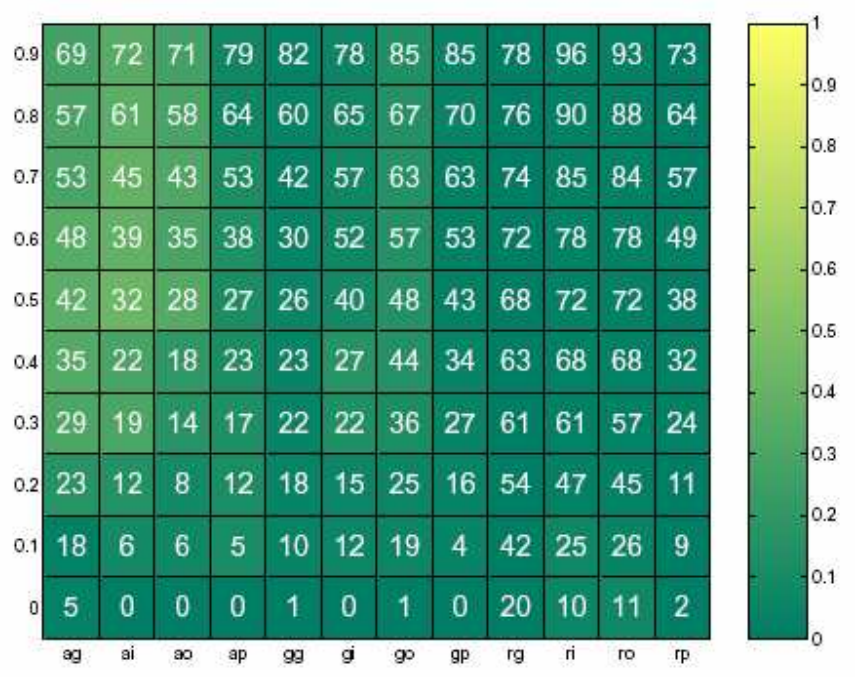
14. ábra Átlagolt QA különbség értékek 0.8-as küszöbnél



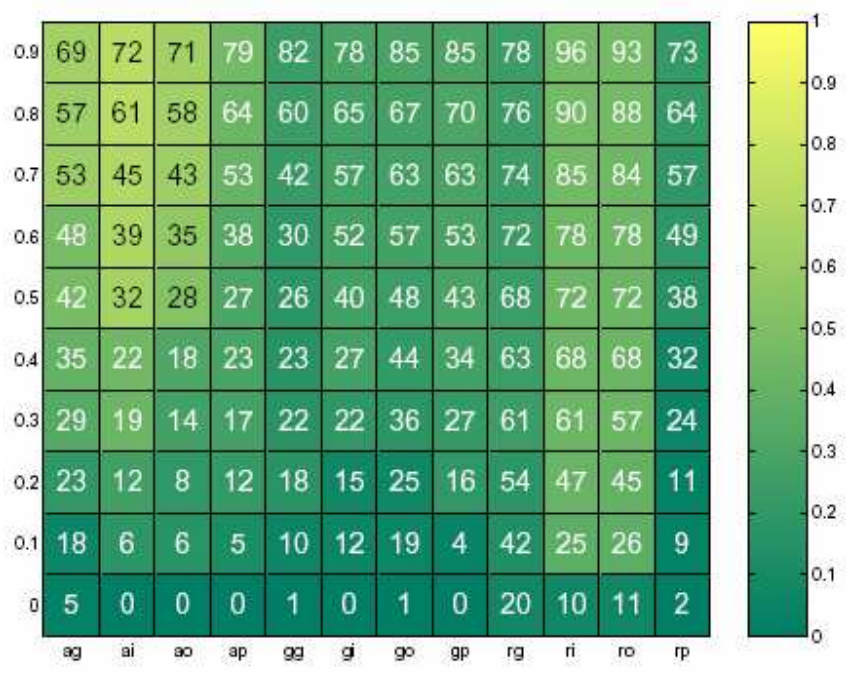
15. ábra Átlagolt QA különbség értékek 0.9-es küszöbnél



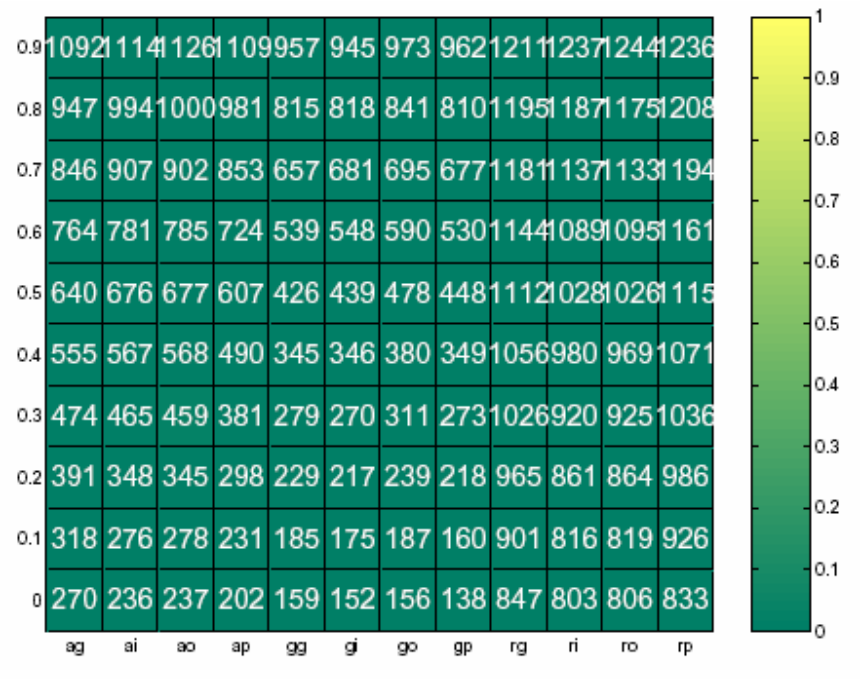
16. ábra Átlagolt QA különbség értékek 0-ás küszöbnél



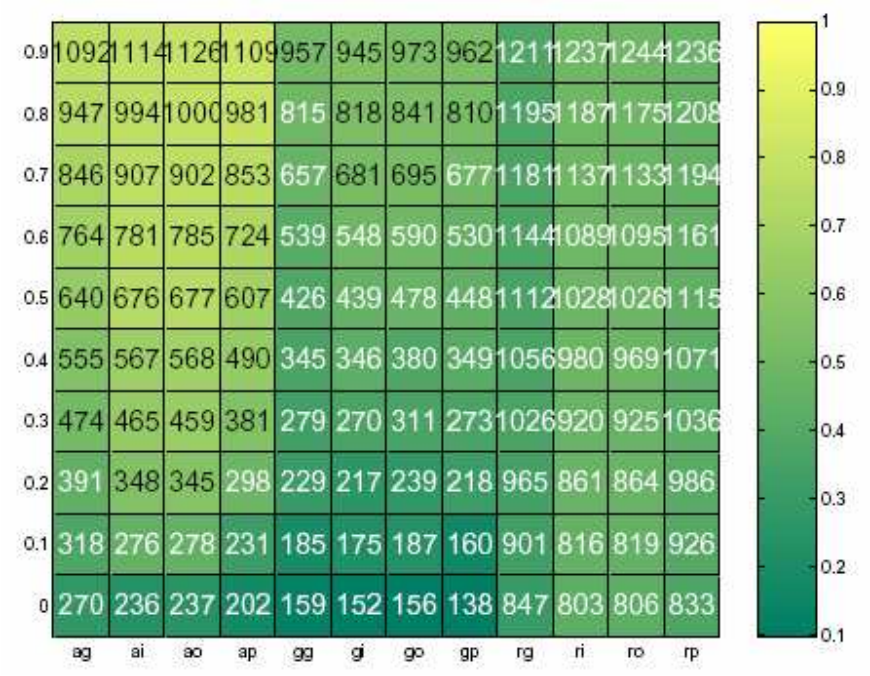
17. ábra Összegzett MI kulcsszavak ÉS kapcsolatban



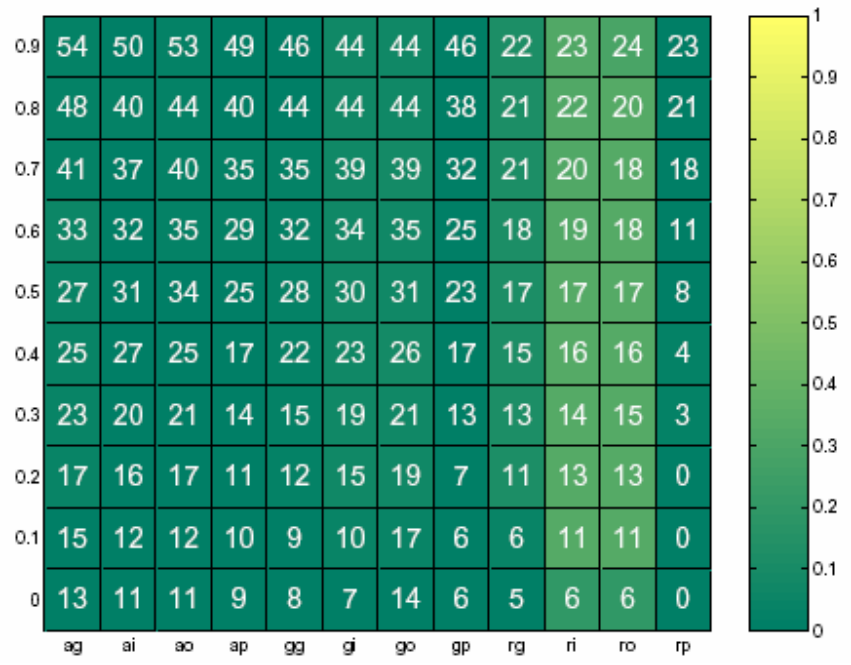
18. ábra Összegzett MI kulcsszavak VAGY kapcsolatban



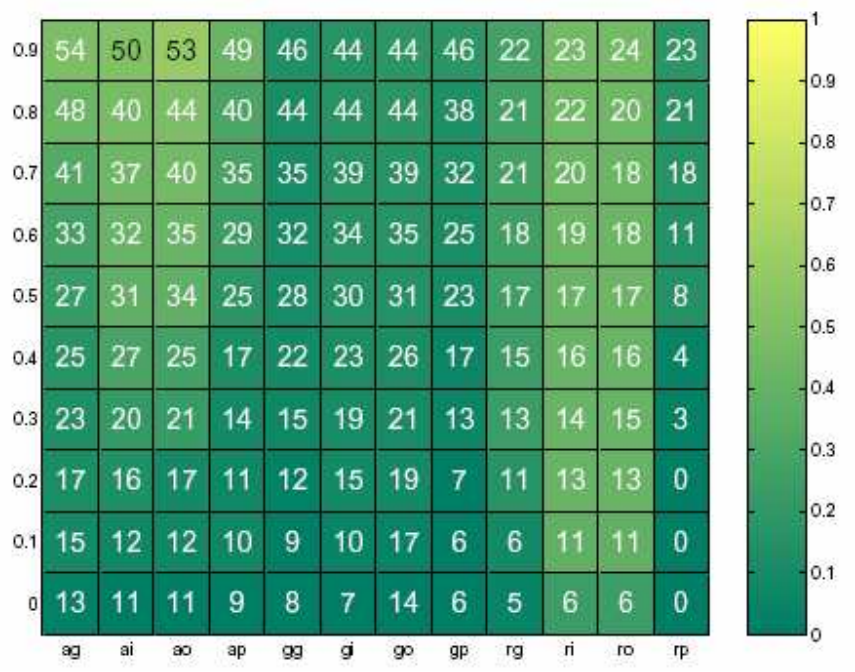
19. ábra Összegzett META kulcsszavak ÉS kapcsolatban



20. ábra Összegzett META kulcsszavak VAGY kapcsolatban



21. ábra Összegzett kezdőlap kulcsszavak ÉS kapcsolatban



22. ábra Összegzett kezdőlap kulcsszavak VAGY kapcsolatban

7 Irodalomjegyzék

1. R. Albert and A.L. Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74:47–91, 2002.
2. A.L. Barabási, R. Albert, and H. Jeong. Scale-free characteristics of random networks: The topology of the world wide web. *Physica A*, 281:69–77, 2000.
3. D.L. Boley. Principal direction division partitioning. *Data Mining and Knowledge Discovery*, 2:325–244, 1998.
4. C.W. Clark and M. Mangel. *Dynamic State Variable Models in Ecology: Methods and Applications.*Oxford University Press, Oxford UK, 2000.
5. V. Csányi. *Evolutionary Systems and Society: A General Theory of Life, Mind, and Culture*. Duke University Press, Durham, NC, 1989.
6. M. Diligenti, F. Coetzee, S. Lawrence, C. Lee Giles, and M. Gori. Focused crawling using context graphs. In *26th International Conference on Very Large Databases, VLDB 2000*, Cairo, Egypt, 10–14 September 2000. http://www.neci.nec.com/lawrence/papers/ focus-vldb00/focus-vldb00.ps.gz.
7. J.M. Fryxell and P. Lundberg. *Individual Behavior and Community Dynamics.* Chapman and Hall, London, 1998.
8. Thorsten Joachims. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In Douglas H. Fisher, editor, *Proceedings of ICML-97, 14th International Conference on Machine Learning*, pages 143–151, Nashville, US, 1997. Morgan Kaufmann Publishers, San Francisco, US.
9. G. Kampis. *Self-modifying Systems in Biology and Cognitive Science: A New Framework for Dynamics, Information and Complexity*. Pergamon, Oxford UK, 1991.
10. J. Kennedy, R.C. Eberhart, and Y. Shi. *Swarm Intelligence*. Morgan Kaufmann, San Francisco, USA, 2001.
11. J. Kleinberg and S. Lawrence. The structure of the web. *Science*, 294:1849–1850, 2001.
12. I. Kókai and A. Lőrincz. Fast adapting value estimation based hybrid architecture for searching the world-wide web. *Applied Soft Computing*, 2:11–23, 2002.
13. A. Lőrincz, I. Kókai, and A. Meretei. Intelligent high-performance crawlers used to reveal topic-specific structure of the WWW. *Int. J. Founds. Comp. Sci.*, 13:477–495, 2002.
14. J. Rennie, K. Nigam, and A. McCallum. Using reinforcement learning to spider the web efficiently. In *Proc. 16th Int. Conf. on Machine Learning (ICML)*, pages 335–343. Morgan Kaufmann, San Francisco, 1999.
15. R. Sutton. Learning to predict by the method of temporal differences. *Machine Learning*, 3:9–44, 1988.
16. R. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, 1998.