Intelligens adatbányászat

Diplomamunka

Gyenes Viktor
Eötvös Loránd Tudományegyetem,
Programtervező matematikus hallgató

Témavezető:

Dr. hablil Lőrincz András
ELTE IK Információs Rendszerek Tanszék

2005. április 21.

Tartalomjegyzék

1. Bevezetés

Ma, az információ korában fontos szerepet tölt be az információ megkeresése, az újdonságok felfedezése. Kétségtelenül a legjobb hely a keresésre az Internet, amely mára a világ legnagyobb adatbázisává fejlődött. Mérete már régen meghaladta azokat a korlátokat, amelyeket az ember a gépek segítsége nélkül fel képes dolgozni. Természetesen születtek keresési módszerek, amelyek a gépek segítségére támaszkodva megkönnyítik a keresést (Google [5], AltaVista [1]). Azonban ezek a hagyományos keresőrendszerek is kezdenek nehézkesen használhatóvá válni, mivel a bennük lévő egyszerű algoritmusok képtelenek az adatok mélyebb analizálására, így nehéz a találatokat a felhasználó igényeihez igazítani.

Például a Google-ben ha megadunk egy keresendő szót vagy szókapcsolatot, akkor a rendszer még jó esetben is nagy valószínűséggel több ezer oldalt ad vissza, aminek nagyon nagy hányada számunkra érdektelen. Emellett esetleg elmulaszt visszaadni olyan oldalakat, amelyek számunkra megfelelőek lennének, mivel szavakban fogalmaztuk meg a kérdésünket, esetleg más szavakat használtva, mint amiket a kívánt oldalak használnak.

Ebből a példából jól látszik, mi is okozza a nehézségeket a feladatban: az emberi nyelv használata (a hatalmas adatmennyiségekkel történő manipulálás ennél kisebb problémának tűnik, hiszen ezt a mai rendszerek is viszonylag jól kezelik, például a Google mára már több mint 8 milliárd oldalt indexel [17]). Az oldalak tartalmát eredetileg embereknek szánták, emberi nyelven íródtak. Ezen kívül a keresésekre vonatkozó kérdéseket is emberek teszik fel, emberi nyelven. Az emberi nyelv viszont tele van többértelmű elemekkel. Ugyanazt a fogalmat több szóval is kifejezhetjük, ugyanazt a mondanivalót több mondatkonstrukcióval is leírhatjuk, ugyanekkor egy szó vagy mondatkonstrukció többfélét is jelenthet, attól függően hol használjuk.

Két törekvés látszik kibontakozóban, amely a fent említett problémákra próbál megoldásokat találni:

- Az egyik az Internet mai állapotára építve a természetes nyelven íródott szövegeket próbálja minél jobban feldolgozni számítógépek segítségével. Mivel a természetes nyelvek feldolgozása már az intelligens információ feldolgozás kora előtt is csábító kihívást jelentett mind a nyelvészek, mind a mesterséges intelligencia kutatók számára, ezek a módszerek nagyobb múltra tekintenek vissza.

- A másik megközelítés az Internet egy új formájának vízióján alapszik, amelyet Semantic Web [8] néven emlegetnek. A Semantic Web központi újítása az adatok olyan tárolása, hogy az könnyen feldolgozható

legyen számítógépekkel, mégis közel maradjon az emberi fogalomreprezentációhoz. Ehhez a mostanában nagy lendülettel terjedő XML [4] és a rá épülő fogalomreprezentációs szabványok, ún. ontológia leíró nyelvek szolgálnak alapul (RDF(S) [7], DAML+OIL [3] [6], OWL [10]). Az ontológiák fogalmakat, és a köztük lévő kapcsolatokat írják le, szemantikus gráfok formájában.

Természetesen ezen törekvések nem ortogonálisak egymásra, hanem egymást kiegészíthetik. A jövőben valószínűleg mindkét irányvonal módszereire, azok kombinációira lesz szükség a hatékony és intelligens adatbányászat megvalósítására. Az ontológiák elegánsan képesek reprezentálni az információ hatékony megtalálásához szükséges háttértudást. Azonban az ontológiákban megjelenő fogalmak elnevezései is az emberi nyelvből származnak, így feldolgozásukhoz elengedhetetlen lesz az természetes nyelvfeldolgozási módszerek alkalmazása. E terület nagy kihívásainak számít például különböző emberek által készített ontológiák egymásnak megfeleltetése [22] [13], összefésülése [45] [44], ontológiák automatikus készítése [40] szövegadatbázisok segítségével.

Dolgozatomban azon irányvonallal foglalkozom, amely a természetes nyelvfeldolgozás segítségével közelíti meg az intelligens adatbányászatot. Áttekintem a természetes nyelvfeldolgozás azon problémáit, megoldásait és eredményeit, amelyek az adatbányászatban is fontos szerepet kapnak, kiemelve és részletesebben tárgyalva néhányat, amelyekből saját munkám során ötleteket merítettem. Ezután ismertetek egy jórészt általam kifejlesztett módszert, amelynek célja ismeretlen szavak magyarázása ismert szavak, lehetséges szinonímák segítségével. Majd végül kitekintést nyújtok egy olyan keresőrendszer terve felé, amely megerősítéses tanulás segítségével megpróbál a felhasználó keresési igényeihez igazodni.

2. Természetes nyelvfeldolgozás adatbányászati szempontból

Az intelligens adatbányászat célja, hogy olyan dokumentumokat találjunk, amelyek a felhasználó számára relevánsak. A jelenlegi keresőrendszerek működése tipikusan a megadott kulcsszavak dokumentumokban történő előfordulásán alapszik. Általában szavak formájában adhatjuk meg, hogy mit is keresünk: mely szavak forduljanak elo a dokumentumokban, mely szavak ne, mely szavak forduljanak elo együtt, egymás mellett, egymás közelében, stb. Azonban a szavaknak több jelentése van, és ugyanazt a dolgot megfogalmazhatjuk más szavakkal is, ezért egyáltalán nem biztos, hogy az ilyen módszerekkel megtalált dokumentumok számunkra megfelelőek lesznek. Túl

sok találat esetén fáradságos lehet a hamis találatok között a valóban fontos dokumentumok kiszűrése az ember számára.

Ezért fontos, hogy a felhasználó pontosabban tudja megfogalmazni mit is keres, illetve, hogy a keresőrendszer értelmezni tudja mind a keresést elindító kérdést, mind a potenciális dokumentumokat. Azonban természetes nyelven írt szöveget (emberi szinten) értelmezni nagyon nehéz feladat. Viszont valószínűleg egy jókora lépéssel közelebb jutunk a megoldáshoz, ha meg tudjuk állapítani, hogy egy adott szövegben a szavak milyen jelentésben szerepelnek, illetve, hogy hogyan kapcsolódnak egymáshoz. Az adatbányászatban használatos természetes nyelvfeldolgozási (Natural Language Processing, NLP) módszerek zöme ezt próbálja megcélozni.

Milyen problémák is merülnek fel az NLP terén a szavakkal kapcsolatban?

- Szavak csoportosítása (osztályozás, klaszterezés): melyek azok a szavak amelyek jelentésükben közeliek, esetleg szinonímák? Hogyan rendezhetők a szavak által reprezentált fogalmak hierarchiába, mik az általános és mik a speciális fogalmak?
- Egy adott szót melyik jelentésében használja az adott dokumentum? (Word Sense Disambiguation)
- Kulcsszó kivonatolás: melyek egy adott dokumentum legfontosabb szavai, hogyan kapcsolódnak egymáshoz? Hogyan foglalható össze egy dokumentum jelentése? (Keyphrase Extraction)

A következőkben megemlítek néhány széles körben használt alapgondolatot és adatbázist a szavakkal és jelentésekkel kapcsolatban, majd áttekintem a fent említett NLP problémákra született megoldásokat, nagyobb hangsúlyt fektetve azokra, amelyek saját munkám előzményeinek és tekinthetők, inspirálták azt.

2.1. Szavak és jelentések - a kontextus

Egy alapvető nyelvészeti feltevés, hogy a szavak jelentését a használati környezet határozza meg. Ugyanazon szó két különböző környezetben különböző jelentéssel bírhat, és két különböző szó, amit hasonló környezetben használhatunk valószínűleg jelentésükben is egymáshoz közeli. Ez a két momentum nagyon fontos az adatbányászat szempontjából. Ugyanazt a gondolatot különböző szavakkal is leírhatjuk, és ugyanazt a szót különböző fogalmak jelölésére is használhatjuk, tehát nem elég csupán a szavak lexikális formájára támaszkodni. Meg kell tudni különböztetni egy szó különböző használatait, illetve fel kell ismerni két különböző szó hasonlóságát. Ennek alapja az előbbi

feltevés szerint lehet a szó kontextusa. Ez a fejezet a kontextus lehetséges definícióira és reprezentációira tér ki.

A szavak használati környezetének megállapítására tipikus módszer, hogy egy megfelelően nagy és reprezentatív szövegadatbázist végigolvasva, valamilyen környezet definíció alapján megkeressük a szavak környezetét. Egy szó használati környezetét többféleképpen definiálhatjuk. Például megadhatjuk, hogy a szó milyen más szavak közelében fordult elő gyakran. Egy másik lehetőség, hogy egy nagy szövegadatbázist szakaszonként tekintünk (például bekezdésenként, vagy fájlonként), és megadjuk, hogy egy szó mely szakaszokban fordult elő. Természetesen egy pontosabb jellemzését adhatjuk meg a környezetnek, ha nem csak azt adjuk meg, hogy egy szó milyen környezetekben fordult elő, hanem azt is, hogy hányszor fordult elő az egyes környezetekben, vagyis elofordulási frekvenciákat gyűjtünk. A következőben felsorolok néhány, az irodalomban használt környezet definíciót:

- Szavak előfordulása egymás közelében (near): egy adott szó környezetének tekintjük az összes olyan szót, aminek a közelében előfordul egy szövegadatbázisban, pontosabban, ha egy rögzített méretű szövegablakban együtt előfordulnak. Az együttes előfordulásokhoz frekvenciákat is rendelhetünk.

- Szavak együttes illeszkedése egy sablonra: ebben az esetben adott egy vagy több sablon, és megszámoljuk, hányszor fordul elő két szó, úgy, hogy megfelel a sablonnak. Ilyen sablon lehet például, hogy hányszor fordulnak elő főnevek együtt felsorolásban, vagy az *és* illetve a *vagy* szavakkal elválasztva.

- Nyelvtani kapcsolat szavak között: a szöveg mondatain végezhetünk nyelvtani elemzést, és megadhatjuk, hogy mely szavak szerepelnek együtt nyelvtani kapcsolatban (például egy cselekvés és annak tárgya, egy főnév és a jelzője). Itt szintén megjegyezhetjük az előfordulási frekvenciákat, de akár a nyelvtani kapcsolat típusát is.

- Szavak szövegszakaszokhoz viszonyított elofordulása: feljegyezhetjük, hogy az egyes szavak mely paragrafusokban (vagy fájlokban) hányszor fordulnak elő.

2.1.1. Környezetek reprezentációja

A környezetek reprezentációjára két kézenfekvő módszer adódik

- Vektortér reprezentáció: a vektortér reprezentációban minden szóhoz egy rá jellemző vektort építünk (*feature vektor*). Ezek a vektorok lehetnek egyszeru skalár vektorok (az i. szó a j. szóval milyen gyakran fordult elő együtt), de tartalmazhatnak absztraktabb elemeket is, például nyelvtani kapcsolat típusa, és elofordulási gyakoriságból alkotott párokat is (az i. és a j. szó milyen nyelvtani kapcsolatban, milyen gyakran fordultak elő).

- Gráfos reprezentáció: ebben az esetben a szavakat egy gráf csúcsainak feleltetjük meg, és élt húzunk be két csúcs között, ha van köztük valamilyen szemantikus kapcsolat, az éleket pedig a kapcsolatot leírásához szükségesen címkézhetjük. Például az együttes elofordulási gyakoriságot reprezentáló élsúlyokat adhatunk meg, vagy megnevezhetjük a csúcsokhoz tartozó szavak közti relációt, amit az adott él reprezentál.

A kétféle reprezentáció természetesen könnyen megfeleltethető egymásnak. Azonban a mögöttük levő különböző matematikai modellek különböző algoritmikus módszereket vetnek fel a szavak és a környezetek elemzésére. Például a vektortér reprezentáció kézenfekvő módszert ad szavak környezetének (és ezáltal maguknak a szavaknak) az összehasonlítására vektorok koszinusz távolságának mérésével. A gráfos reprezentáció pedig fontos gráfelméleti eredmények vonatkoztathatók a szavak hálózatára.

2.1.2. Skálamentes kisvilág gráfok

A közelmúltban Watts és Strogatz kezdeményezése nyomán a gráfok egy új osztályának tanulmányozása indult el [47] [46] [29], [30] [36] természetes módon kialakuló hálózatok modellezésének céljából. Watts és Strogatz *kisvilágoknak* nevezte el őket az úgynevezett kisvilág jelenség miatt [37]: ha tekintjük az emberek hálózatát, és két embert az ismeretség köt össze, akkor azt kapjuk, hogy két tetszőleges ember átlagosan 6 ismerősön keresztül van összekötve. Vagyis az így kapott gráf, amely az embereket világát írja le, kicsi átmérőjű; kicsi a világ!

A kisvilág gráfok kedvező tulajdonságokkal rendelkeznek: hierarchikus szervezésűek, és a rácsokhoz hasonlóan magas a klaszterezettségük, azonban a véletlen gráfokhoz hasonlóan rövid átlagos úthossz jellemzi őket, vagyis kicsi az átmérőjük. A magas klaszterezettség azt jelenti, hogy nagy a valószínusége annak, hogy egy csúcs szomszédai egymásnak is szomszédai lesznek. Sok esetben egy másik tulajdonság is megfigyelhető a kisvilág gráfokban, nevezetesen, hogy a csúcsok fokszáma hatványeloszlás szerinti, vagyis viszonylag kevés nagy fokszámú csúcs van bennük, és sok kis fokszámú. A nagy fokszámú csúcsok szerepe a gráf távolabbi klasztereinek összekötése, emiatt

lesz tetszőleges két pont között mindig rövid út. A gráf átmérője a csúcsok számának logaritmusával arányosan nő, innen a skálamentes elnevezés.

Watts és Strogatz megmutatták, hogy több természetes módon kialakult hálózat, az USA elektromos hálózata, a *Caenorhabditis elegans* nevű féreg idegrendszere, vagy az amerikai színészek együttes film-szereplési hálózata is ilyen jellegű gráfot alkot. Barabási és Albert az Internet skálamentes szerkezetét vizsgálták [12], míg Ferrer és mások [26] megmutatták, hogy a szavak egy mondaton belüli együttes elofordulása alapján épített gráf is skálamentes kisvilágot alkot, amiből szintén következik, hogy vannak szavak, amiknek jól meghatározott jelentéseik vannak, másoknak pedig összekötő szerepük van a nyelvben.

A tény, hogy a nyelv szavai kisvilág gráfot alkotnak elméleti alapokat ad a szavak környezeteit gráfként reprezentáló algoritmusok számára. Ezek alapján különböző gráf klaszterező eljárásokkal a szavak jelentéscsoportokba sorolhatók, egy szónak több jelentése felfedezhető, és a környezet alapján arra is következtethetünk, hogy a szó egy adott előfordulása melyik jelentésének a megnyilvánulása.

2.2. Nyelvi adatbázisok

A szavak jelentésével kapcsolatos NLP algoritmusok segítésére több, manuálisan vagy fél-automatikusan összeállított adatbázis létezik. Ezek közül hármat említenék meg, amelyeket munkám során használtam. A WordNet az angol nyelv szavait jelentésekbe tömörtő hálózat, a British National Corpus nagy mennyiségű angol nyelvű szövegeket tartalmazó adatbázis, a SemCor pedig kisebb méretű, ám ún. szemantikusan jelölt szövegadatbázis.

2.2.1. WordNet

A WordNet [23, 11] egy gazdag lexikális adatbázis, mintegy 150.000 angol szót, szókapcsolatot és szófordulatot tömörít 110.000 jelentésbe. A jelentéseket szóhalmazok segítségével definiálja, ezeket szinoníma halmazoknak nevezi. Egy szinoníma halmaznak egy vagy több eleme lehet, amelyek olyan szavak amik által a jelentés egyértelművé válik; egy szinoníma halmaz jelentése a benne szereplő szavak jelentéseinek metszete. Egy szó természetesen több szinoníma halmazba is tartozhat, így nyilvánul meg a szavak többértelműsége.

Ezen kívül a WordNet kapcsolatokat definiál mind a szavak, mind a szinoníma halmazok között. A legfontosabb kapcsolatok a hiperníma és hiponíma relációk, amik a számítástechnikában jól ismert *is-a* relációnak, és annak inverzének felelnek meg. Segítségükkel a WordNet a fogalmakat hier-

archiába rendezi. Minél általánosabb egy fogalom, annál feljebb kerül a hierarchiában. Egyéb relációkra példa a rész-egész reláció fogalmak között, vagy az ellentét reláció. Szavak közti reláció például a szavak alaki hasonlósága, például egy igei és egy főnévi alak között.

A WordNet egy általános, tudományágtól független hálózat, mely manuálisan lett összeállítva több év alatt folyamatos finomítással. Sok NLP algoritmus használja segítségként vagy referenciaként saját teljesítményének mérésére a szavak jelentésének osztályozása terén.

2.2.2. British National Corpus

A British National Corpus (BNC) egy 100 millió szavas szövegadatbázis, amely az angol nyelv minden használati területéről próbál szövegmintákat szolgáltatni. A BNC mind írott, mind beszélt nyelvi forrásokból tartalmaz szövegkivonatokat. A szövegek SGML formátumú szövegfájlokban vannak eltárolva, és sokféle hasznos információ is mellékelve van hozzájuk, melyek számítógéppel is könnyen feldolgozhatóak. Adottak például a paragrafus és mondathatárok, illetve a szavak szófaja. A szavak szófaját automatizált eszközökkel határozták meg (Claws tagger [24, 2]).

A BNC (illetve más hasonló méretű adatbázisok) segítségével statisztikák készíthetők szavak előfordulási frekvenciáiról, jellemző használati környezetek kereshetők az egyes szavakhoz.

2.2.3. SemCor

A SemCor egy olyan kézzel elkészített szövegadatbázis, amelyben a szavak mellé adottak azok jelentése az aktuális környezetben, vagyis minden szóhoz adott, hogy a WordNet-ben felsorolt jelentései közül épp melyikben fordul elő.

A SemCor-hoz hasonló adatbázisok segítségével megbecsülhetők a szavak WordNet jelentéseinek eloszlása, illetve az, hogy az egyes jelentések milyen valószínűséggel reprezentálódnak egy adott szóval.

2.3. Szavak csoportosítása jelentéseik szerint

A természetes nyelven írott szövegek megértéséhez fontos lépés a szavak többértelműségének feloldása, vagyis annak megállapítása, hogy mely szavak hasonló jelentésűek, mely szavak szinonímák.

Habár a WordNet rendelkezésre áll az angol nyelv szavaira, mégis fontosak lehetnek olyan algoritmusok, amelyek automatizálni képesek a szinonímák

meghatározásának folyamatát. Egyrészt, egy WordNet jellegű adatbázis manuális összeállítása nagyon sok munkát igényel (a WordNet jelenlegi formáját mintegy 15 év alatt érte el). Másrészt egy általános célú adatbázis nem képes lefedni minden szakterületet kellő részletességgel, míg a más esetekben túl finom jelentéseket különböztet meg. Például a WordNet a *computer* szóhoz egy olyan jelentést is hozzárendel, ami azt jelenti, hogy 'egy ember, aki számol', míg a *dialog* szónak hiányzik belőle a számítástechnikában gyakran használt, felhasználói felülettel kapcsolatos jelentése.

A fent említett okok miatt sok módszer született a szavak jelentésekbe tömörítésére. Ezen algoritmusok általában a szavak környezetét használják fel a szavak jelentéseinek megkülönböztetésére, és a szavakat klaszterekbe osztják vagy a szavakat jellemző vektorok osztályozásával, vagy szavak gráfjának klaszterezésével. A klaszterezéssel egyidejűleg az egyes klaszterek hierarchiába is szervezhetők. A szavak automatizált jelentésekbe tömörítésével olyan jelentéseket fedhetünk fel, amelyek egy adott szakterületre jellemzőek.

2.3.1. Pantel és Lin klaszterező eljárása

Pantel és Lin megoldása [41] [42] a szavak vektor-tér reprezentációját használja a szavak hasonlóságának eldöntésére. A szavak feature vektorában minden bejegyzés egy lehetséges használati környezetre vonatkozik, ahol a használai környezet egy másik szót jelent. Azt mondják, hogy egy szónak egy másik szó a környezete, ha valamilyen szövegben előfordulnak nyelvtani kapcsolatban egymással. Például a *piros alma* kifejezésben a *piros* az *alma* jelzője, így a *piros* az *alma* egy környezete. A környezet értéke a szó és a környezet kölcsönös információja (Pointwise Mutual Information) lesz.

Egy w szó c környezetben történő előfordulásának gyakoriságát $F_c(w)$-vel jelölve a kölcsönös információt a következőképpen számolják:

$$mi_{w,c} = \frac{\frac{F_c(w)}{M}}{\frac{\sum_i F_i w}{M} \times \frac{\sum_j F_c j}{M}} \qquad ahol \qquad M = \sum_i \sum_j F_i(j)$$

a szavak összesített együttes előfordulási frekvenciája. Ezt felhasználva a szavak hasonlóságát a a feature vektoruk koszinusz távolságaként definiálják. Jelölje f_w a w szóhoz felépített feature vektort: $f_w = (mi_{w,c_1}, ..., mi_{w,c_n})$. Ekkor a w_i ás a w_j szó hasonlósága:

$$sim(w_i, w_j) = \frac{< f_{w_i}, f_{w_j} >}{|f_{w_i}||f_{w_j}|}$$

Az algoritmus ezután három fázisból áll. Az első fázisban kiszámolják minden szóra a k hozzá leghasonlóbb szót. A kísérleteikben $k = 10$

volt. Aztán a második fázisban szoros klasztereket határoznak meg, ahol az egyes klaszterek tagjai úgynevezett magokat (comittee) alkotnak. Az algoritmus annyi magot próbál meghatározni, amennyit csak lehet, amellett a feltétel mellett, hogy egy újonnan létrehozott mag nem lesz túl hasonló a már meglévő magokhoz. A harmadik fázisban minden szót hozzárendelnek a hozzá leghasonlóbb klaszermagokhoz, így kapva meg a szó különböző jelentéseit.

A klaszterek meghatározása következőképpen történik: először minden szóra a hozzá leghasonlóbb szavakat klaszterezik átlag-link klaszterezéssel. Az így kapott klaszterekre kiszámolnak egy pontszámot, amely a klaszteren belüli konzisztenciát méri. Minden szóhoz a legmagasabb pontszámú klasztert tartják meg. Aztán a klasztereken pontszám szerint csökkenő sorrendben végighaladva kiszámolják a klaszterek centroidját a frekvencia vektoraik átlagolásával és a fent említett kölcsönös információ számítási módszerrel. Ha a centroid hasonlósága az eddig megkapott centroidok valamelyikéhez kisebb egy adott küszöbnél, akkor a centroidot megtartják, különben eldobják. Ezután minden szóra megnézik, hogy van-e olyan centroid, amihez egy adott küszöbnél közelebb van. Ha vannak olyan szavak amelyekre ez nem teljesül, akkor az ilyen szavakra rekurzívan folytatják az eljárást. Így végül olyan klasztereket kapnak, amik minden szót lefednek abban az értelemben, hogy egy szó legalább egy klaszter centroidjához az adott küszöbnél közelebb van.

A magok meghatározása után a következőt végzik el minden szóra: a szót ahhoz a klaszterhez rendelik, aminek a centroidja a legközelebb van a szó feature vektorához. Ezután kitörlik azokat a feature-öket a szó feature vektorából, amik átfednek a centroid feature vektorával, és újra megkeresik a legközelebbi klasztert. Ezt addig iterálják, amíg a legközelebbi klaszter egy adott köszöbnél közelebb van. Így egy szóhoz több klasztert, azaz több jelentést is hozzá rendelnek, és a szó ritka jelentéseit is képesek felfedni, amiben kulcsszerepet játszik az, hogy a már felfedezett jelentések feature-it kitörlik a szó feature vektorából.

Az algoritmus hatékonyságának mérésére a kapott jelentéseket a szavak WordNet jelentéseivel hasonlították össze. Azt mérték, hogy az algoritmus által adott jelentések hány százaléka felel meg a szó egy WordNet jelentésének, illetve, hogy a szó WordNet jelentései közül hányat talált meg az algoritmus. Természetesen a klaszterek és a WordNet szinoníma halmazainak összehasonlítása nem triviális. Ehhez először a WordNet hierarchia alapján a szinoníma halmazok hasonlóságát definiálják, majd egy szó és egy szinoníma halmaz hasonlóságát adják meg a szó jelentésein keresztül, és ennek segítségével egy klaszter és egy szinoníma halmaz hasonlóságára is adnak egy márőszámot. Ha egy szóra kapott klaszter és a szó egy WordNet-beli szinoníma halmazának hasonlósága meghalad egy adott küszöböt, akkor korrektnek minősítik

az algoritmus által megadott jelentést. A további részletek a [41]-ben találhatók. Eredményül azt kapták, hogy az algoritmusuk által adott jelentések 60 százaléka megfeleltethető WordNet jelentésnek, és a WordNet jelentések 50 százalékát meg is találta az algoritmus.

Pantel egy későbbi munkájában [42] a kapott jelentás csoportokat cimkékkel látta el, amelyek összefoglaló, általános neveket adnak a csoportokban szereplő szavaknak. Például cégneveket tartalmazó halmazhoz a *cég* címkét rendeli. Ennek segítségével a jelentések hierarchiába szervezhetők, a szavak közti *is-a* reláció részben megállapítható. Ez fontos lehet természetes nyelvekkel dolgozó keresőrendszerek számára, mivel gyakran előfordul, hogy egy kérdésben egy általános fogalom szerepel, a válasz pedig a fogalom egy speciális előfordulásának megtalálásával kapható meg, vagy esetleg fordítva.

2.3.2. Dorow gráf klaszterezés alapú megoldása

Dorow és mások a szavak kontextusának gráf reprezentációját használták a szavak kategorizálása során [14]. Abból indultak ki, hogy a szövegekben felsorolásokban szereplő főnevek között szoros szemantikus kapcsolat áll. Egyszerű reguláris kifejezések segítségével olyan főnév párokat kerestek a BNC-ben, amelyeket az *and*, *or* szavak vagy vessző választ el. Ez alapján egy gráfot építettek, amelynek csúcsai a főnevek voltak, és két főnév akkor lett összekötve, ha együtt előfordultak az előbb említett típusú felsorolásokban.

A felépített gráf segítségével a szavak jelentéscsoportjainak megtalálása (például a hét napjai, vallásformák, stb.), illetve egy-egy szó több jelentésének felfedése volt a céljuk. Dorow [14]-ben amellett érvel, hogy a szavak jelentéseinek száma, és a gráfban kiszámolt klaszterezési együttható között erős negatív korreláció van, erősebb, mint a pozitív korreláció a szavak jelentéseinek száma, és a szavak frekvenciája, vagy gráfban kiszámolt foka között. Vagyis a magas klaszterezettségű szavaknak kevés jelentése van, így ezek alkalmasak egy-egy jelentéscsoport reprezentációjára, míg az alacsony klaszterezettségű szavak többjelentésű szavak lesznek. Emiatt a gráf *gyenge* klaszterezését végzik, ami azt jelenti, hogy az alacsony klaszterezettségű szavakat több stabil klasztermaghoz is hozzákapcsolják.

Ehhez két gráfklaszterezési eljárást használnak, a klaszterezési együttható alapú klaszterezést, és a Markov klaszterezést [52]. A klaszterezettségi együttható egy gráfban a nódusok egy jellemzője, ami a nódus környezetének összekötöttségét méri. Egész pontosan annak a mérőszáma, hogy egy nódus szomszédainak hányad része van összekötve:

$$c(w) = \frac{\text{azon háromszögek száma amiben w szerepel}}{\text{azon háromszögek száma, amiben w szerepelhetne}}$$

A klaszterezettségi együttható alapú klaszterezés egyszerű eljárás, aminek lényege, hogy az összekötő szerepet játszó, alacsony klaszterezettségű szavak törlésével a gráf olyan részekre esik, amely részeken belül nagy a kohézió a szavak között (magas klaszterezettségűek). Ehhez egyszerűen minden szóra kiszámolják a klaszterezettségi együtthatót, és amelyek nem érnek el egy adott küszöböt, azt kitörlik a gráfból. Az ezután összekötve maradt komponensek adják a klasztermagokat. A klasztermagok megtalálása után minden klasztermaghoz hozzáveszik azokat a szavakat, amelyek az eredeti gráfban kapcsolódtak a klasztermag elemeinek valamelyikéhez. Így egy-egy alacsony klaszterezettségű szó több klaszterbe is belekerülhet, megtalálva ezzel a szó több jelentését.

A van Dongen által kifejlesztett Markov klaszterezés (MCL) [52] a gráfban véletlen sétákat szimuláló klaszterezési eljárás. Az alapötlet az, hogy a véletlen séták egy magas klaszterezettségű részeken sok időt töltenek el, mielőtt kijönnének onnan, átugorva egy másik klaszterbe. Az algorimus hatékonyan implementálható mátrixszorzások segítségével. Az algoritmus részletei a [52]-ben találhatók.

A Markov klaszterezés olyan klaszterezést ad, ahol minden nódus csak egy klaszterbe tartozik. A probléma megkerülése érdekében Dorow és társai a szó gráf egyfajta duális gráfját klaszterezik, és az ott kapott klaszterezés megfelel a szógráf egy gyenge klaszterezésének. A duális gráf egy olyan gráf amelynek minden csúcsa az eredeti gráf egy élének felel meg, és a duális gráfban két csúcs között akkor megy él, ha az eredeti gráfban a csúcsoknak megfelelő élek szerepelnek egy háromszögben. Az így kapott duális gráf egyértelműbb lesz, így könnyebb megtalálni a klsztereket. Ezzel a módszerrel a tulajdonképpen az eredeti gráf éleit klaszterezik, ami egyszerűen megfeleltethető a csúcsok egy gyenge klaszterezésének.

Az eredmények manuális kiértékelése után azt találták, hogy mindkét módszer képes volt a olyan klaszterek megtalálására, melyek elemei szoros jelentésbeli kapcsolatban vannak egymással, és egy egy szó több ilyen klaszterbe is tartozhat, amelyek megefelelnek a szó különböző jelentéseinek.

2.4. Szavak aktuális jelentésének megállapítása

A szinonímák illetve az egyértelmű jelentések meghatározása mellett egy másik fontos lépés a szövegek megértésében annak eldöntése, hogy egy többjelentésű szót mikor melyik jelentésében használunk. E probléma megoldására vállalkozó algoritmusok felhasználhatják a szavakat jelentésekbe csoportosító algoritmusok kimenetét, vagy a szavak egy manuális csoportosítását. A feladat egy szó aktuális előfordulásához hozzárendelni egyet a lehetséges jelentései közül. Ennek alapjául általában a szavak lokális környezete szolgál

a szövegben, de használható más, külsö forrás is, például enciklopédiák, manuálisan összeállított információ források a szavak jelentéséről.

Mivel a jelentések megkülönböztetésének problámája az egyik legrégebbi, ami foglalkoztatja a nyelvészeket, egy sereg módszer született ennek megoldására az 50-es évektől kezdődően. A módszerek sokfélesége miatt lehetetlen mindet még csak felületesen bemutatni is. Ide és Véronis munkájukban [39] kimerítő és részletes képet adnak a WSD algoritmusok történetéről és különféle változatairól.

A WSD algoritmusok teljesítményének kiértékelésére egy automatizált módszer az algoritmus futtatása egy jelentésekkel jelölt korpuszon (mint például a SemCor), és az eredmény összehasonlítása a korpusz által megjelölt jelentéssel. Azonban, mivel a WordNet-hez hasonló lexikális adatbázisok túl finom jelentéseket is megkülönböztetnek, gyakran összevonják a környezet alapján megkülönböztethetetlen jelentéseket, vagy más módszereket alkalmaznak a kiértékelések relaxálására.

A legtöbb modern statisztikai WSD algoritmus [25] [16] [18] azon az ötleten alapul, hogy egy szó két előfordulásában ugyanazt jelenti, ha ugyanabban a kontextusban használják. Ezért különféle gépi tanulási és osztályozási technikákat használva manuálisan jelentésekkel felcimkézett adatbázisokon megtanulják, hogy egy-egy szó mely rá jellemző környezetben, melyik jelentését reprezentálja. Tehát ezek az algoritmusok egy adott szó jelentéseit a szó korábbi előfordulásai alapján próbálják meg megkülönböztetni, aminek az a hátránya, hogy a szó sok előfordulása szükséges, mire egy osztályozó képes megtanulni a megfelelő osztályozást, ráadásul azokra a szavakra, amik nem fordultak elő a tanítás során, nem tud semmit sem mondani az osztályozó.

Lin munkájában [33] a következő elvet használja: két *különböző* szó hasonlót jelent, ha ugyabban a lokális környezetben fordulnak elő. Tehát egy szó jelentésének feloldására más, masonló környezetben előforduló szavak jelentését használja. A módszer előnye, hogy nem igényel egy jelentésekkel jelölt korpuszt, és minden szóra egy külön osztályozót, hogy megtanulja megkülönböztetni egy-egy szó jelentéseit. Ehelyett minden szóra ugyanazokat az információ forrásokat használja. Ennek segítségével ismeretlen szavak jelentésének magyarázását is képes lehet elvégezni a rendszer.

A rendszer a WordNet-et használja a szavak lehetséges jelentéseinek felsorakoztatására. A szavak lokális kontextusát pedig nyelvtani függőségi relációk (például cselekvés tárgya, jelzős szerkezet, stb.) segítségével definiálja. Ehhez egy nagy hatékonyságú nyelvtani elemzővel [34] elemez végig egy 25 millió szavas Wall Street Journal korpuszt.

Az algoritmus egy lokális kontextus adatbázison alapul, melyben minden szóhoz felsorakoztatja az elemzés során hozzá talált lokális kontextusokat, vagyis, hogy milyen nyelvtani szerkezetekben, milyen szavakkal, hányszor

fordult elő. Egy szó aktuális jelentésének megállaptása a következőképpen történik: az input szöveg elemzésével megkeresi a kérdéses szó lokális környezeteit. Majd a lokális kontextus adatbázisban megkeresi azon szavakat, amelyek előfordultak a kérdéses szóval azonos lokális kontextusban; ezt a kérdéses szó szelektorainak nevezi. Egy olyan jelentést választ a kérdéses szónak, amely maximalizálja a szó és szelektorainak hasonlóságát. Ezt a lépést egy a jelentések között a WordNet hierarchia segítségével definiált hasonlósági mérték segtségével teszi. További részletek a [33]-ben találhatók. Intuitíven azt mondhatnánk, hogy egy olyan jelentést választ, amely legközelebb áll a szelektorok lehetséges jelentéseinek metszetéhez.

Lin az algoritmusát a SemCor adatbázison tesztelte, ami a találati feltétel relaxálása nélkül 56%-osan, a találati feltétel relaxálásával pedig 70% felett teljesített.

2.5. Kulcsszó-kivonatolás

Egy fontos feladat a dokumentumok feldolgozása során a dokumentumok csoportosítása, annak eldöntése, hogy mely dokumentumok tartoznak ugyanabba a kategóriába.

Egy lehetőség e probléma megközelítésére a dokumentumok kulcsszavakkal (illetve több szavas kifejezésekkel) címkézése. Ezután a dokumentumokhoz tartozó kulcsszavak segítségével már könnyebb feladat megállapítani, hogy melyik dokumentum melyik kategóriába tartozik.

A kulcsszó keresésnek két formáját különböztetik meg; a kulcsszó hozzárendelést, és a kulcsszó kivonatolást. A kulcsszó hozzárendelés esetén egy előre megadott kulcsszó listából rendelünk a dokumentumhoz egy vagy több kulcsszót, ahol a dokumentumnak nem feltétlenül kell tartalmaznia a hozzárendelt kulcsszavakat. A kulcsszó kivonatolás esetén a kulcsszavak a dokumentum szövegéből kerülnek ki. Mindkét módszer mellett fel lehet sorakoztatni érveket és ellenérveket is, mégis a gyakorlatban az irodalomban talált módszerek a másodikat részesítik előnyben, mivel abban az esetben a teljes folyamat könnyebben automatizálható, nem szükséges egy jól megválasztott kulcsszó lista hozzá.

Annak ellenőrzése, hogy egy-egy algoritmus helyes kulcsszavakat talál-e, nem triviális feladat. Két lehetőség adódik: összehasonlítani a szerzők által megadott kulcsszavakkal, vagy szubjektíven megítélni, hogy a kapott kulcsszavak valóban relevánsak-e. Az első előnye, hogy automatizálható, és nem szubjektív, a különböző algoritmusok számszerűen összehasonlíthatóak. Viszont gyakran még az emberek sem értenek egyet abban, hogy mik a jó kulcsszavak, így gyakran nem mindenki találja megfelelőnek még azt sem, amit a szerző megad. Emiatt érdemes lehet úgy pontozni, hogy egy adott algorit-

mus által szolgáltatott kulcsszavak közül megmondjuk mik azok, amelyeket a legtöbb ember elfogad, és mik azok amelyeket sokan rossznak gondolnak.

Turney [49] egy összefoglalást ad az irodalomban megtalálható, munkáját megelőző módszerekről a kulcsszó kivonatolás terén. Bemutatása szerint az addigi módszerek részben heurisztikus [31], részben felügyelet nélküli tanulást alkalmazó algoritmusok [38]. A heurisztikák legtöbbször a szintaktikán alapultak, mint például nagybetűs vagy dőltbetűs szavak, a dokumentum vagy a fejezetek címében használt szavak, stb. megdaása kulcsszavakként. Legtöbbször egy nagy és zajos kulcsszó listát eredményeznek, amik kevéssé feleltek meg a szerzők által megjelölt kulcsszavaknak.

Turney egy újszerű, felügyelt tanulási algoritmust fejlesztett ki a kulcsszavak keresésére. A módszer lényege, hogy szerzők által megadott kulcsszavakon tanulva, valamilyen felügyelt tanulási algoritmus segítségével a rendszer a szavak különféle paraméterei alapján megtanulja a szavak két kategóriába sorolását: kulcsszó, vagy nem kulcsszó. A tanítás után a rendszer más adatbázison is alkalmazható kulcsszavak keresésére.

A GenEx [49] elnevezésű rendszerük két modulból áll, egy genetikus algoritmusból, és egy kivonatolóból. A kivonatoló benete egy dokumentum, kimenete pedig egy kulcsszó lista. A kivonatoló tizenkét paramétere befolyásolja a döntést, hogy mit tekint a rendszer kulcsszónak. A paraméterek a szavak szótövesítésére, minimális hosszára, gyakoriságára, stb. vonatkoznak. Ezen paraméterek hangolására szolgál a genetikus algoritmus, melynek célja, hogy a példa dokumentumokon a kulcsszó kivonatoló minél inkább olyan kulcsszavakat adjon, mint amilyeneket a szerzők határoztak meg. A tanítási fázis után a genetikus algoritmus modulra már nincs szükég, újabb dokumentumok esetén a kulcsszó kivonatolás már gyorsan megy. A tesztek szerint a rendszer az emberek által megadott kulcsszavak mintegy 29%-át megtalálta, és az emberi olvasók bírálata szerint a talált kulcsszavak mintegy 80%-a elfogadható.

Turney módszerének mintájára Frank és mások kifejlesztettek egy hasonló, felügyelt tanuláson alapuló rendszert [20], amit Kea-nak (Keyphrase Extraction Algorithm) neveztek el. A módzserük naív Bayes osztályozót használ annak megtanulására és eldöntésére, hogy mely szavak a kulcsszavak. A naív Bayes osztályozó egy egyszerű és népszerű tanuló algoritmus, amely a Bayes formula alkalmazásán alapul, azzal a feltevéssel, hogy az egyes paraméterek (változók) függetlenek egymástól (ezért hívják naívnak), ám a kísérletek szerint abban az esetben is használhatóan működik, amikor ez a feltevés nem teljesen igaz. Egy bővebb áttekintés igényében a [19]-at ajánlom az érdeklődő olvasó figyelmébe.

A modell eredetileg két paramétert használ: egy kifejezés $TF \times IDF$ pontszámát, és a kifejezés *távolságát*. A $TF \times IDF$ mérték egy gyakran

használt mérték az adatbányászat terén, és azt méri, hogy egy adott szó mennyire jellemző egy dokumentumra:

$$TF \times IDF(W, D) = P(\text{egy szó D-ben az épp W}) \times$$
$$-logP(\text{W egy dokumentumban előfordul})$$

Egy kifejezés *távolsága* egyszerűen a kifejezés első előfordulásának pozícióját jelenti, vagyis, hogy milyen távol van a kifejezés első említése a dokumentum kezdetétől.

E két paraméter függvényében az osztályozó megtanulja besorolni a szavakat a kulcsszavak illetve a nem kulcsszavak közé. A modellt kiterjesztették egy harmadik, tárgyterület függő paraméterrel is, amely azt méri, hogy egy adott szó hányszor fordul elő más dokumentumokban a szerző által megadott kulcsszavak között. Ezzel a kiterjesztéssel a rendszer jobban képes tárgyterület függő kulcsszavak megtalálására.

Frank és társai összehasonlították rendszerüket Turney-évl, és azt kapták, hogy a két rendszer teljesítménye körülbelül megegyezik, ha tárgyterülettől függetlenül alkalmazzák. Azonban a Frank és társai által bemutatott módszer tanítása jóval kevesebb időt igényel, ezért egy új tárgyterületre evaló betanítása sokkal könnyebb. Emellett a tárgyterület specifikus információ figyelembe vételével a teljesítménye jobb lesz az általános GenEx-nél.

Később Turney egy munkájában kiterjesztette a Kea algoritmust [51] néhány további paraméterrel, ami a kapott kulcsszavak közötti koherenciát hivatott mérni. A paraméterek kiszámolásához webes keresési technikákat használt AltaVista lekérdezések formájában. A kiterjesztéssel sikerült az algoritmus által adott rossz kulcsszavak egy részét kiszűrni, azáltal, hogy azok jelentésben nem illeszkedtek a többi kulcsszóhoz.

3. Korpusz alapú neurális hálózat módszer ismeretlen szavak magyarázására

3.1. Bevezetés

A munka egy felügyelet nélküli algoritmus mutat be, ismeretlen szavak WordNet synsetekkel történő magyarázására. A módszer egy hibrid rendszernek hívható, mivel korpuszból nyert statisztikai információkat, és lexikális adatbázisokat egyaránt használ.

Az Internet egy hihetetlenül nagy szövegadatbázis; nagy mennyiségű téma specifikus szövegeket találhatunk benne, amelyek gyakran tartalmazhatnak ismeretlen jelentésű szavakat. A rendszerünk célja, hogy megmagyarázzuk

azokat a szavakat, amiket nem ismerünk, vagy ismereteink szerint nem illik az adott szövegkörnyezetbe, viszont az adott jelentésben nagy mennyiségű szöveg áll róla rendelkezésre. Ebben az esetben a módszerünk olyan WordNet-beli szinoníma halmazokat keres, amelyek jelentésükben közel állnak a szó ismeretlen jelentéséhez.

A módszer azon a jól bevált elven alapszik, hogy a szavakat jelentését a használatuk környezete nagyban meghatározza [35]. Ez az ötlet azt sugallja, hogy két szó jelentésbeli hasonlóságát az együttes előfordulásuk valamilyen mértékével lehetne definiálni - például a főnevek felsorolását vizsgálva [15], kölcsönös információ segítségével (PMI-IR [50]). A szavakat jelentésükben közelinek mondhatjuk, ha gyakran fordulnak elő egymás közelében. Vannak azonban olyan módszerek is, amelyek másodrendű együttes előfordulásokon alapulnak: akkor mondják, hogy két szó jelentésükben közeli, ha gyakran fordulnak elő ugyanabban a környezetben, például vannak olyan szavak, amelyekkel mindketten gyakran előfordulnak együtt [21]. Ezen módszer az utóbbi elven alapul.

A módszer három információforrást használ. A szavak lehetséges jelentéseit a WordNet szinoníma halmazainak felhasználásával állapítja meg, az egyes szavak jelentés-eloszlásának megbecslésére pedig a SemCor adatbázist használja. A szavak együttes előfordulási statisztikáit pedig a BNC szövegadatbázisból nyeri.

A fent említett adatokat felhasználva egy rekonstrukciós hálót terveztünk, amely megpróbálja megkeresni azon jelentéseket, amelyek használati környezete hasonlít az ismeretlen szóéra.

A módszert egy 80 TOEFL(Test Of English as a Foreign Language [9]) szinoníma kérdésből álló teszten értékeltük ki. Az ismeretlen szót szimulálandó, a kérdés szóra vonatkozó, jelentéssel kapcsolatos információkat nem vettük figyelembe. A rendszer a kérdések 76.25%-ára, azaz 61 kérdésre tudott helyesen válaszolni. Összehasonlítás képpen, egy felmérés szerint az amerikai felsőoktatási intézményekbe jelentkező nem angol anyanyelvű diákok átlagosan a TOEFL szinoníma kérdések 64.5%-ára válaszolnak helyesen [32]. Ugyanezeken a kérdéseken tesztelve, a klasszikus LSA [32] algoritmus 64.4%-osan, Turney PMI-IR [50] algoritmusa 73.75%-osan teljesített. Terra és Clarke korpusz alapú módszere [48], amely egy terabájtnyi web adatbázist használt, 81.25%-ot ért el. Ezzel szemben a mi módszerünk csak a 700 megabájtnyi BNC adatbázist használta. Jarmasz és Szapakovicz egy szótár alapú rendszert készített [28], amely 78.75%-osan teljesített ezeken a kérdéseken. Az adatbázisok méretbeli és minőségbeli különbségei miatt az eredmények nem összehasonlíthatóak egy-az-egyben.

Különböző modulok összefésülésével lehetőség nyílik egy robosztus és hatékony rendszer kifejlesztésére. Két összefésülési lehetőség a szakértők

szorzata, és a szakértők keveréke módszer, amelyekben a paraméterek hangolására a megerősítéses tanulás segíthet [?]. Turney [43] a szakértők szorzata módszer adaptálásával 97.5%-os eredményt ért el a TOEFL kérdéseken. Ezt a magas pontszámot egyenként kevésbé sikeres modulok hatékony összefésülésével érte el. A mi algoritmusunk az ilyen rendszerekben egy lehetséges modulként fogható fel.

3.2. A TOEFL teszten kipróbált módszerek

Turney PMI-IR algoritmusa [50] felfogható együttes előfordulásokon alapuló módszerként, azonban az egész Internetet használta adatbázisként, az AltaVista kereső segítségével. Az algoritmus kölcsönös információt használ a szavak hasonlóságának mérésére. Az AltaVista egy speciális lekérdező operátorát, a NEAR operátort használja, amely olyan dokumentumokat ad vissza, amely az argumentumként megadott két szót egymáshoz közel, egy 10-es méretű ablakon belül tartalmazza. Így megkapható azon oldalak száma, amely a szavakat egy contextusban tartalmazza.

A Turney cikkjében leírt lekérdezések közül a következő bizonyult a legjobbnak a TOEFL kérdések megoldása során:

$$score(c_i) = \frac{hits((p \,\mathrm{NEAR}\, c_i)\,\mathrm{AND\,NOT}((p\,\mathrm{OR}\, c_i)\,\mathrm{NEAR}\,\text{"}not\text{"}))}{hits(c\,\mathrm{AND\,NOT}(c\,\mathrm{NEAR}\,\text{"}not\text{"}))},$$

ahol p a kérdés szó, c_i az i-edik lehetséges válasz, és *hits* pedig a pedig azt jelzi, hogy az AltaVista hány oldalt talált, ami az argumentumként kapott lekérdezésnek megfelelt. A pontszámok kiszámolása után minden lehetséges válasz szóra, azt a választotta, amely a legnagyobb pontszámot kapta.

Landauer LSA [32] algoritmusa szintén szavak együttes előfordulásán alapult. A szavak $TF \times IDF$ (Term Frequency $\times$ Inverse Document Frequency) vektorait gyűjtötte ki egy enciklopédiából, ahol minden szövegrészletnek egy cikk felelt meg az enciklopédiában. A $TF \times IDF$ mérték azt méri, hogy egy adott szó mennyire jellemző egy dokumentumra:

$$TF \times IDF(W, D) = P(\text{egy szó D-ben az épp W}) \times$$
$$-logP(\text{W egy dokumentumban előfordul})$$

Így egy mátrixot kapott, amelynek sorai a szavakra vonatkoztak, oszlopai pedig a dokumentumokra. Ezután a mátrix dimezióját sajátérték dekompozíció segítségével 300-ra csökkentette. Ezen előfeldolgozás után a szavak hasonlóságát a csökkentett mátrix megfelelő sorainak koszinusz távolságával

tudja mérni. Landauer a módszerét egy memória modellként fogta fel, ahol a dimenzió csökkentés az általánosítási képességet hivatott szolgálni.

Terra és Clarke [48] többféle együttes előforduláson alapuló statisztikai módszert hasonlított össze egy egy terabájt méretű korpuszon. Turney PMI módszerével jobb eredményt értek el mint Turney maga, valószínűleg azért, mert a saját korpuszon több lehetőségük volt a lekérdezések implementálása terén, például kipróbálhattak több kontextus ablak méretet (míg Turney csak az AltaVista által kínált 10-es ablakot használhatta).

Jarmasz és Szapakovicz [28] más módszerrel próbálkoztak. A Roget's Thesaurus nevű lexikális adatbázist használták hogy a szavak közötti szemantikus relációk gráfjában két szó közötti utak hosszát határozzák meg, a szavak jelentésbeli hasonlóságának mérésére.

Turney legfrissebb eredményei [43] a TOEFL tesztekkel kapcsolatban azt mutatják, hogy egyenként nem túl sikeres modulok kombinálásával jelentősen javíthatók az eredmények. Három különböző kombinálási szabályt próbált: a keverés, a logaritmikus és a szorzat módszer. A szorzat módszer bizonyult a leghatékonyabbnak közülük.

3.3. Az algoritmus

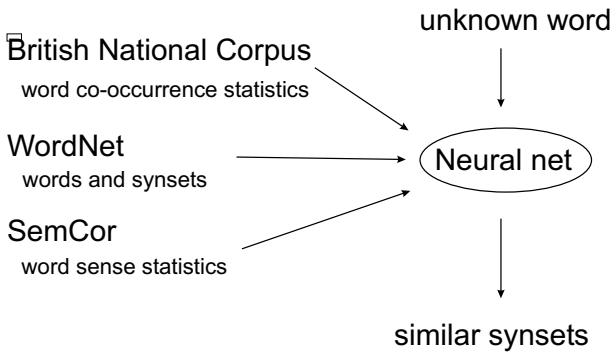
3.3.1. Információ források

Mint már említettem, az algoritmus három adatbázist használ. A szavak együttes előfordulására vonatkozó adatokat a BNC adatbázisból nyeri. Kihasználja a szavak szófaj cimkézését is, azonban enélkül is megoldható lett volna, mivel a BNC-t is automatikusan cimkézték fel a CLAWS szófajcimkéző [24] segítségével. Az algoritmus külön szöként kezeli egyazon szó két különböző szófajú előfordulását. Például a *work* szó főnévként és igeként két külön szónak számít.

A szavakat szótövesítettük a WordNet disztribúcióban lévő szótövesítő szolgáltatást használval. Ezen kívül a WordNet-ből a módszer csak a szavak jelentésekbe tömörítését használja ki.

A módszernek a következő becslésekre is szüksége van: ha adott egy jelentés, mint szinoníma halmaz, mely szót milyen gyakorisággal (illetve valószínűséggel) használjuk a jelentés kifejezésére? Illetve, ha adott egy szó, a lehetséges jelentései közül melyikben milyen gyakorisággal (illetve valószínűséggel) fordul elő? Ezen statisztikák elkészítésére van szükség a SemCor adatbázisra.

Mivel a fent említett információk minden szóra szükségesek voltak, csak olyan szavakkal foglalkoztunk, amelyek mindhárom adatbázisban alőfordul-



1. ábra. **A rendszer sematikus rajza**

tak. Így állapítottuk meg a felhasznált 23141 szót és a hozzá tartozó 22012 jelentést. A szűk keresztmetszetet a SemCor adatbázis jelenti, mivel az egy relative kicsi adatbázis. Később kiterjesztettük a szavak listáját olyan szavakra is, amelyek nem szerepeltek a SemCor adatbázisban, viszont a rájuk vonatkozó statisztika egyértelmű volt: azon szavakra, emelyeknek csak egy jelentésük van, és a jelentésüket reprezentáló szinoníma halmaz egyetlen elemű (azaz egyedül az adott szóból áll) a fent említett valószínűségek értéke 1. Ezzel a kiterjesztéssel 54572 szóról és 53443 hozzá tartozó jelentésről volt meg a kellő információnk.

3.3.2. Szavak közötti együttes előfordulási mérték

Mint ahogy azt a bevezetőben jeleztem, a módszer célja a az ismeretelen szavakhoz jelentésben közeli szinoníma halmaz keresése, ahol az ismeretlen azt jelenti, hogy rendelkezésre áll nagy mennyiségű szöveg, amiben előfordul, de nem tudunk semmit a jelentéséről, vagyis nem szerepel egyik WordNet jelentésben sem. Akkor gondolunk két szót jelentésben közelinek, ha hasonló környezetekben fordul elő.

Annak a valószínűségét, hogy a w_1 szó a w_2 szó közelében fordul elő, becsüljük a következőképpen:

$$P(w_1|w_2) = \frac{f(w_1, w_2)}{f(w_2)},$$

ahol $f(w_1, w_2)$ a w_1 és a w_2 szó együttes előfordulási frekvenciája, egy adott ablakon belül, $f(w)$ pedig a w szó frekvenciája.

Azt mondjuk, hogy w_1 és w_2 egymáshoz közeliek, ha kölcsönösen nagy valószínűséggel fordulnak elő egymás környezetében, vagyis mind $P(w_1|w_2)$, mind $P(w_2|w_1)$ magas. Ezt fejezi ki a következő mérték:

$$N(w_1, w_2) = min(P(w_1|w_2), P(w_2|w_1))$$

Ez a mérték nem érzékeny a gyakori szavakra, mivel ha w_0 egy gyakori szó, akkor minden w szóra a $P(w|w_0)$ érték kicsi lesz, mivel a gyakori szavak nem kötődnek egy adott környezethez. Így nincs szükség az úgynevezett stopszavak (például névelők, segédigék) kiszűrésére, a mérték automatikusan gondoskodik róla.

Ezen közelségi mértéknek a célja a szavak környezetének jellemzése. Egy w szóra kiszámoljuk a közelségét az összes többi szóhoz. Az n legnagyobb közelségű szó alkotja a w szó *környezetét* vagy *kontextusát*. A kíséleteinkben $n = 100$ volt. Ha a w_i szó benne van a w szó kontextusában, akkor a kettejük viszonya az $N(w, w_i)$ értékkel jellemezhető.

A szavak kontextusára vonatkozó információ mátrix alakba írható, ahol a mátrix i. oszlopa az i. szóra vonatkozó kontextus vektort jelent. Jelölje a szavak kontextusaira vonatkozó mátrixot N_W, amely formálisan a mátrix a következőképpen definiálható: $N_W(i, j) = N(w_j, w_i)$

3.3.3. Szavak és szinoníma halmazok

A SemCor adatbázisban minden szó mellé meg van adva, hogy melyik WordNet jelentésben szerepel az adott mondatban. Ezen bejegyzések megszámlálásával a szükséges valószínűségeket megbecsülhetjük. Annak a valószínűsége, hogy egy w szó az s jelentésben szerepel, a következőképpen becsülhető:

$$P(s|w) = \frac{f(s, w)}{f(w)},$$

ahol $f(s, w)$ a w szó gyakorisága az s jelentésben, és $f(w)$ pedig a w szó gyakorisága, bármely jelentésében. Szükség van még arra a valószínűségre is, hogy egy s jelentést egy w szó fejez ki:

$$P(w|s) = \frac{f(w, s)}{f(s)},$$

ahol $f(s)$ az s jelentés gyakorisága, bármelyik szó is fejezi ki. Ezek a valószínűségek szintén összefoglalhatók mátrix alakban. Jelölje a $P(s|w)$ valószínűségekből alkotott mátrixot WS, ahol $WS(i, j) = P(s_j|w_i)$, és hasonlóan, a $P(w|s)$ valószínűségekből alkotott mátrixot SW, ahol $SW(i, j) = P(w_j|s_i)$.

3.3.4. Jelentések közötti együttes előfordulási mérték

A szavak közti közelségi mérték az előbbiek segítségével átvihető a jelentésekre is. Ha adott egy s jelentés, akkor a hozzá közeli jelentések az s jelentést

reprezentáló szavakhoz közeli szavak jelentései lesznek, a megfelelő valószínűségi súlyozással. Ezt fejezi ki a három mértékből alkotott mátrixok, mint leképezések, konkatenációja (az oszlopvektoros jelölés miatt a konkatenációt balról való szorzásként felírva):

$$N_S = WS * N_W * SW$$

3.3.5. Rekonstrukciós hálók működése

A rekonstrukciós típusú mesterséges neuronhálók sematikus modellje a következőképpen képzelhető el: adott két neuron réteg, köztük egy kapcsolati mátrixszal. A hálózat alsó rétege az ún. bemeneti réteg, ahol a háló az inputját kapja. A felső réteg az ún. rejtett, vagy belső reprezentációs réteg, ahol a hálóban kialakul egy belső reprezentáció az inputról a rekonstrukció során.

Matematikailag ez a következőképpen írható le: a hálózat két rétegének két valós (oszlop-) vektor feleltethető meg, amelyek hossza az egyes rétegekben szereplő neuronok számával egyeznek meg, és a vektorban lévő értékek a neuronok aktivitásait hivatottak reprezentálni. A kapcsolati mátrix, ami a kapcsolatok erősségét tartalmazza, egy valós mátrixszal reprezentálható. Ez a mátrix nem kell, hogy négyzetes legyen, a két réteget reprezentáló vektor dimenziója különbözhet. Ekkor az aktivitás vektorok kapcsolati mátrixszal (vagy annak transzponáltjával) történő szorzása a két vektortér közötti transzformációnak felel meg.

A rekonstrukciós háló célja a következő: tekintsük a kapcsolati mátrix oszlopait az input vektorhoz tartozó vektortér bázisvektorainak. A célunk az input vektor optimális rekonstrukciója a bázisvektorok lineáris kombinációjaként, vagyis a lineáris kombinációval valamilyen mérték szerint (például négyzetes eltérés) a lehető legközelebb kerülni az inputhoz. Ekkor a belső reprezentációs rétegben kialakult aktivitások lesznek a lineáris kombináció együtthatói, ez adja a belső reprezentációt.

A hálót példa bemeneteken taníthatjuk különféle tanítási szabályok segítségével (ICA, NMF), hogy a kapcsolati mátrixban található súlyokat hangolva a megfelelő bázisvektorokat megtaláljuk [1]. Bizonyos tanítási szabályok esetén fontos szempont az is, hogy a bemeneteket lehetőleg kevés bázisvektor segítségével tudjuk rekonstruálni, vagyis ún. ritka reprezentációt tudjunk kialakítani

[1] Ezek a bázisvektorok nem feltétlenül lesznek a tér algebrai értelemben vett bázisvektorai, mivel nem garantált, hogy lineárisan függetlenek lesznek. A cél csupán az, hogy olyan vektorokat találjunk, melyek lineáris kombinációja esetén a háló kellő általánosítási képességgel rendelkezik majd, vagyis olyan bemeneteket is képes legyen rekonstruálni, amelyekkel a tanulás során nem találkozott

(Sparse Code Shrinkage).

A mi esetünkben a hálót nem vetettük alá tanításnak, ehelyett explicit módon megadtuk a bázisvetorokat. A fent bemutatott, jelentések kontextusát leíró mátrixokat használva, egy jelentéshez (mint báziselemhez) tartozó bázisvektor a jelentés kontextusát reprezentáló vektor lesz. Így az ismeretlen szó kontextusát az ismert jelentések kontextusaival próbáljuk meg rekonstruálni.

Egy adott kapcsolati mátrix és input vektor esetén a belső reprezentáció a következő költség függvény minimalizációjaként írható fel:

$$J(a) = |x - Qa|_{L_2} + \alpha S(a)$$

ahol Q a kapcsolati mátrix, x az input vektor, a a rejtett reprezentációs vektor, $S(a)$ pedig egy a ritkább reprezentációt előnyben részesítő tag, α ritkítási együtthatóval. Esetünkben $S(a) = |a|$.

Az optimalizációs problémára egy direkt megoldás adható (a problémát leíró differenciál-egyenletből származtatva) a következő formulával:

$$a = \left(Q^T Q + \alpha I\right)^{-1} Q^T x,$$

ahol I az identitás mátrix. Egy gradiens módszerből származtatott iteratív megoldás is adható a következő frissítési szabály segítségével:

$$\Delta a = Q^T(x - Qa) - \alpha a$$
$$a = a + \eta \Delta a.$$

Mindkét megoldásnak vannak előnyei és hátrányai is. A direkt módszer pontos megoldást ad, viszont kiszámolásához jelentős memóriára lehet szükség nagy mátrix esetén, még akkor is, ha az ritka, emellett az invertálás műveletigénye is igen nagy. Az iterációs megoldás kevés memóriát igényel, viszont a gradiens módszer használata csak lineáris konvergenciát biztosít.

3.3.6. A módszerben használt rekonstrukciós háló

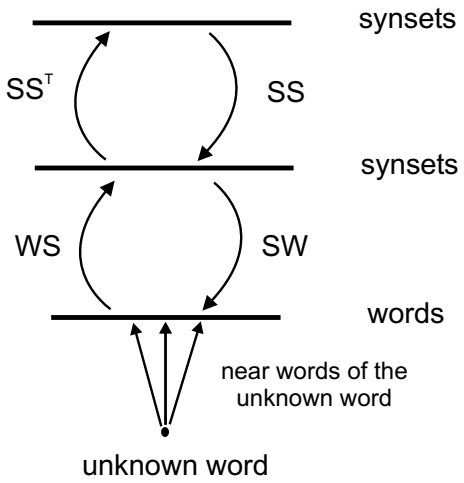
Az ismeretlen szó jelentésének azonosítása céljából egy kétszintű rekonstrukciós hálót terveztünk. A háló bemenete az ismeretlen szó környezetét reprezentáló vektor. Az első szint feladata az ismeretlen szó környezetében szereplő szavak jelentéseinek meghatározása. A második szint bemenete az első szint belső reprezentációja, és feladata a azon jelentések megtalálása, amelyek az ismeretlen szó környezetét rekonstruálják.

Az első szint alsó rétegében a neuronok szavakat reprezentálnak, a felső rétegben levő neuronok pedig a hozzájuk tartozó jelentéseket. Tehát az alsó

rétegben 23141 neuron található, a felsőben pedig 22012. A kapcsolati mátrix itt két részre bomlik, egyik a felülről-lefelé transzformációt írja le, a másik az alulról-felfelé transzformációt. Ezek a mátrixok rendre a már korábban tárgyalt SW illetve WS mátrixok [2].

A második szint alsó rétege az első szint felső rétegével azonos. A második szint felső rétegében is jelentéseket reprezentáló neuronok vannak (szintén 22012), ezek a lehetséges jelöltek a bemenet jelentésére. Itt a két réteg közötti transzformációkat a N_S mátrix írja le.

Optimális rekonstrukció esetén a felső rétegben megjelenő aktivitásokat a lehetséges jelentések és bemenet által reprezentált jelentés azonosságának mértékeként értelmezzük. Egy nagy aktivitás azt jelenti, hogy az adott jelentés kontextusa nagy mértékben vett részt az ismeretlen szó kontextusának rekonstruálásában. Tehát azokat a jelentéseket választjuk az ismeretlen szó valószínű jelentésének, amelyekhez tartozó aktivitások a legnagyobbak.



2. ábra. **A két szintű rekonstrukciós háló architektúrája:** Az első szint input rétegében az i. neuron aktivitása $N(w_i|w)$, ahol w az ismeretlen szó. A kapcsolati mátrixok definíciója a következő: $WS(i,j) = P(s_j|w_i)$, $SW(i,j) = P(w_j|s_i)$, $N_S = WS * N_W * SW$, ahol $N_W(i,j) = N(w_j, w_i)$. Optimális rekonstrukció esetén a legfelső réteg aktivitásait vizsgáljuk. A legnagyobb aktivitáshoz tartozó jelentéseket tekintjük az ismeretlen szó valószínű jelentéseinek.

[2] A mátrixok majdnem egymás transzponáltjai, ugyanott vannak bennük nem nulla értékek, de a nem nulla értékek, az átmeneti valószínűségek nem egyeznek meg

Abban az esetben, ha az ismeretlen szó jelentéséről valamilyen információ áll rendelkezésre (például a szó szófaja), a jelölt jelentésel listája csökkenthető, tehát csak azokat a jelentéseket vesszük figyelembe a rekonstrukció során, amelyekben érdekeltek vagyunk. Ezen szűrés a rendszerben igen egyszerűen megvalósítható, hiszen csak annyit kell tennünk, hogy a második szint felső rétegében csak azokat a jelentéseket szerepeltetjük, amelyekben érdekeltek vagyunk. Ezt a szűrést felülről-lefelé (Top-Down, TD) megszorításnak nevezzük.

3.4. Tesztek

3.4.1. Teszt környezet: TOEFL

A módszert egy 80 TOEFL kérdésből álló szinoníma teszten próbáltuk ki. Egy kérdés a következőképpen nézett ki:

grin? exercise, rest, joke, smile

A feladat, hogy kiválasszuk a négy lehetséges válasz szó közül azt, amelyik a legjobban hasonlít a kérdés szóra, ami itt a *grin*. Jelen esetben a megoldás a *smile*.

Ahhoz, hogy a mi módszerünkkel oldjuk meg a feladatot, a kérdések előfeldolgozására van szükség. Először is meg kell határozni a szavak szófaját, majd a szófajnak megfelelően szótövesíteni kell a szavakat. Ezt a WordNet segítségével tettük meg. Amennyiben a szófaj eldöntése nem egyértelmű, akkor azt a szófajt állapítjuk meg minden szóra egy kérdésben, amelyik a legtöbbre ráillik. Ha egy kérdésben minden szóra több szófaj is lehetséges (például főnévkén és igeként is értelmezhető mindegyik), akkor a kérdést megdupláztuk, és ennek feloldásával egy későbbi fázisban foglalkoztunk.

A teszt esetén a kérdés szó játszotta az ismeretlen szó szerepét, ezért a kérdés szónak nem kellett benne lennie minden adatbázisban, csupán a BNC-ben, hogy megfelelő kontextus információnk legyen róla. Mivel a BNC tartalmazott minden szót a TOEFL teszttel kapcsolatban, ez nem jelentett problémát. Annak érdekében, hogy a rendszer a kérdés szót tényleg ismeretlen szóként kezelje, a statisztikáinkból eltávolítottunk minden kérdés szóra vonatkozó információt.

Az előfeldolgozások elvégzése után a kérdés szó környezetére mint bemenetre lefuttattuk a rekonstrukciós hálót. Ezután minden válasz szóhoz hozzárendeltünk egy súlyt, ami a szó jelentései közül a legnagyobb aktivitású jelentés aktivitásával egyezett meg. Majd a legnagyobb súlyú szót választottuk megoldásként a kérdésre.

Abban az esetben, ha egy kérdést meg kellett duplázni a nem egyértelmű szófaj miatt, a rendszer mind a két kérdést megoldotta, és azt a megoldást

választotta a végleges megoldásnak, ahol az első két legnagyobb súly aránya nagyobb volt (vagyis ahol egyértelműbbnek gondolta a választ).

3.4.2. Kontroll feladat

Annak érdekében hogy megállapítsuk, hogy megérte-e a szinoníma halmazok, és ezzel együtt a jelentések alkalmazása, készítettünk egy olyan rekonstrukciós hálót is amely csak szavakat használt.

A hálónak csak két rétege van. Az alsó ugyanaz, mint az eredeti háló legalsó rétege. Ezzel szemben a felső réteg neuronjai itt szavakat reprezentálnak. A kapcsolati mátrix a már korábban definiált N_W mátrix, amelynek oszlopaia a szavak kontextusait reprezentálják. A háló bemenete az eredetihez hasonlóan az ismeretlen szó környezete. A rekonstrukció során a felső réteg magas aktivitású szavai lesznek azok, amelyeket az ismeretlen szó jelentésével megegyezőnek tekintünk.

3.4.3. Teszt esetek

A következő teszt eseteket készítettük el:

Direkt módszer/iteráció. A hálózat első szintjének optimalizációja megoldható direkt módszerrel is, mivel a kapcsolati mátrixok nagyon ritkák, tehát itt csak a direkt módszert alkalmaztuk. A második szinten csak az iterációs módszert lehetett alkalmazni, mivel itt a kapcsolati mátrix sokkal sűrűbb volt. Kipróbáltuk 1, 10, 100 iterációval a háló működését. Sajnos a futtatások igazolták a gradiens módszer lassú, lineáris konvergenciáját.

TD megszorítás. Az összes lehetséges jelentés használata mellett kipróbáltunk két TD megszorítást is. Az egyikben csak azokat a jelentéseket szerepeltettük a rekonstrukcióban, amelyek a kérdés szóval megegyező szófajúak voltak. A másikban csak a négy válasz szóhoz tartozó jelentéseket szerepeltettük a rekonstrukcióban.

Szavak száma. Mind az alap 23141, mind a bővített 54572 szóból álló szókészlettel próbálkoztunk.

3.5. Eredmények

3.5.1. Teszt esetek összehasonlítása

A különböző teszt esetekben adott helyes válaszok számát az 3.5.1 és a 3.5.1 táblázatok foglalják össze. A legjobb eredmény 54572 szó használatával, az

első szinten direkt módszerrel, a második szinten 100 iterációval született; ekor a kérdések 76.25%-ára jól válaszolt a rendszer.

	Teljes	Szófaj	Válaszok
1 iter	58	58	58
10 iter	57	57	57
100 iter	60	58	57
direkt	-	-	56

1. táblázat. **Eredmények 23141 szóval:** A táblázat a TOEFL teszt 80 kérdése adott helyes válaszok számát tartalmazza. Az oszlopok a különböző TD megszorításokra vonatkoznak: a *Teljes* esetén minden jelentés a jelöltek között volt, a *Szófaj* esetben csak a kérdés szó szófajával megegyező jelentések, a *Válaszok* esetben pedig csak a lehetséges válasz szavakhoz tartozó jelentések voltak a jelöltek között. A sorok a különböző optimalizációs módszerekre vonatkoznak a háló második szintjének futtatása során: 1, 10, 100 *iter* az iteráció számra vonatkozik az iterációs megoldás esetén. A *direkt* pedig a direkt megoldási módszerre vonatkozik. Direkt optimalizációra a számítás és memória igénye miatt csak a legkevesebb jelölt jelentést felsorakoztató esetben, vagyis a *Válaszok* TD megszorítás esetén volt lehetőség.

	Teljes	Szófaj	Válaszok
1 iter	60	60	60
10 iter	60	59	60
100 iter	61	57	58
direkt	-	-	54

2. táblázat. **Eredmények 54572 szóval:** Az oszlopok és sorok címkéinek ugyanaz, mint az 3.5.1. táblázat esetén.

Látszik, hogy a TD megszorítások esetén romlott a teljesítmény, pedig az ember úgy gondolná, kevesebb lehetőség közül könnyebb kiválasztani a jót.

A háló bemeneteként szolgáló ismeretlen szó környezete természetesen nagyon zajos. Azt gondoljuk, hogy azért lett rosszabb az eredmény, mert a zaj rekonstruálásához szükség van segítségként a többi jelentés környezetére is. Minél kevesebb jelentést (bázisvektort) használunk a rekonstrukció során, annál nehezebb pontosan rekonstruálni a bemenetet, és ez kihat az együtthatók relatív sorrendjére, vagyis a jelentések súlyára.

Fontos viszont megjegyezni, hogy mind a több szó használata, mind az iterációszám növelése javított az eredményen. Látható, hogy a *Szófaj* TD megszorítás esetén az iterációszám növelése még tudta ellensúlyozni az előbb említett kevesebb bázisvektor használatának problémáját, viszont a *Válaszok* TD megszortás esetében már nem.

A jelentések használata nélküli kontroll feladatot csak a TD megszorítás nélkül esetben futtatuk le, mivel az előző futtatások esetén azok nem okoztak javulást. Az eredményeket a 3.5.1 táblázat foglalja össze. Amint az látszik, a jelentések használata növelte a találatok számát, főleg a kevés iterációs esetekben, azonban a háló képes volt ezt a hátrányt behozni az iterációszám növelésével.

	23141	54572
1 iter	53	54
10 iter	56	56
100 iter	60	59

3. táblázat. **Eredmények a kontrol hálóval:** Az oszlopok a 23141 szavas és az 54572 szavas futtatásokra vonatkozna, a sorok az iterációk számára.

3.5.2. A válaszadás megfontolása

Ha van valami elképzelésünk arról, hogy mennyire vagyunk biztosak a válaszban, akkor eldönthetjük, hogy válaszolunk-e egy adott kérdésre, vagy kihagyjuk. A cél a helyes válaszok arányának növelése, a megválaszolt kérdések arányának rovására. A rendszer eddig minden kérdésre válaszolt, azonban néhány bizonytalan kérdés kihagyásával a találati arány növelhető. Vezessük be a következő két arányszámot:

$$r = \frac{\text{megválaszolt kérdések száma}}{\text{összes kérdés száma}}$$

$$p = \frac{\text{helzes válaszok száma}}{\text{megválaszolt kérdések száma}}$$

Az r jelöli a válaszadási arányt (recall), a p pedig a találati arányt (precision).

Volt néhány lehetséges válasz szó, amely nem szerepelt a SemCor adatbázisban, így ezekre nem voltak meg a megfelelő adatok. Ezekről a rendszer semmit nem tudott mondani, így $-\infty$ súlyt rendelt hozzájuk, és így leghátra

sorolta őket. Sajnos abban az esetben, ha pont a helyes válasz szó volt ismeretlen, természetesen nem adhatott jó választ a rendszer. Ez különösen akkor volt esélyes, amikor több lehetséges válasz szó is ismeretlen volt egy kérdésen belül.

A teszt során minden TOEFL kérdésben legalább két szó ismert volt. Kipróbáltuk hogyan változik a p és az r érték, ha a válaszadást olyan esetekre korlátozzuk, amikor legalább 2, 3 vagy 4 lehetséges válasz szó ismert. Az változást a 3 (a) ábra mutatja. Látható, hogy a kérdések kihagyása növeli a találati arányt.

A lehetséges válaszokhoz rendelt súlyok több információt tartalmaznak, mint egyszerűen a sorrend. Például az első két súly aránya értelmezhető az adott válasz bizonyosságának mértékeként. Sajnos a nagyobb bizonyosság és a helyes válasz nem mindíg járnak együtt ebben az esetben. Ezen bizonyossági mérték alapján egy küszöb adható arra, hogy mikor válaszoljunk egy kérdésre, és mikor hagyjuk azt ki. Ha az arány a küszöb alá esik, a rendszer inkább nem válaszol a kérdésre. A 3 (b) ábra mutatja a p és r értékek változását. A küszöb növelésével 1-től indulva (ami nem jelent megszorítását) 2.4-ig a találati arány folyamatosan nő 91.5%-ig, 60%-os válaszadási arány mellett. Efölé növelve a küszöböt a találati arány már nem nő tovább, csak a válaszadási arány csökken.

Egy másik lehetőség az első két súly arányának vizsgálatán kívül az első súly vizsgálata önmagában. Adhatunk egy küszöböt az első súly értékére, amely alatt a kérdésre nem válaszol a rendszer. Nem meglepő módon így a találati arány 100%-ra növelhető, ám ez a válaszadási arány gyors esésével jár. Ezt a 3 (c) ábra mutatja.

A háló működésének illusztrálására szolgál a következő ábra, mely a legfelső, jelölt jelentéskhez tartozó réteg aktivitásainak változását mutatja az iteráció során az ismeretlennek tekintett *classroom* szóra, mint bemenetre futtatva. Látható, hogy néhány aktivitás kiemelkedik, míg a legtöbb kicsi marad.

3.6. Diszkusszió

Természetesen, a módszerünk hatékonysága nagyban függ a használt adatbázisok méretétől és minőségétől. Mint már a bevezetőben említettem, léteznek az irodalomban hivatkozott egyéb módszerek, amelyek jobban teljesítenek, de az összehasonlítás nem könnyű, hiszen a használt adatbázisok megváltoztatásával az eredmények is jelentősen változhatnak. Két szempontot kell figyelembe venni az összehasonlításkor: a használt korpusz méretét, és a használt lexikális adatbázis szemantikus gazdagságát.

Turney PMI-IR módszere enyhén rosszabbul teljestett mint a miénk, egészen pontosan 73.75%-osan. Ugyanez a módszer Terra web korpuszán már jobban, 81.25%-osan teljesített. Terra cikkjében azt is megmutatja, hogy a corpusz mérete egészen 500 GB-ig nagyban befolyásolja a teljestítményt. A többiekhez viszonyítva az LSA elég gyengén szerepel a maga 64.5%-ával, viszont ki kell emelni, hogy az LSA sokkal kisebb dokumentum halmazon dolgozott, mint a többi módszer. A többi módszerrel ellentétben az LSA-t nem arra tervezték, hogy egy hatékony kérdés megválaszoló rendszer legyen, hanem az emberi memória modelljénem szánták.

Ahogy azt Prof. Landauer elmondta, egy fontos külöbség, hogy az LSA a kérdések ismerete nélkül először végigolvassa a szövegadatbázist, majd elvégzi a dimenzió csökkentést. Ezután, a kérdések feltevése után az LSA azonnal megadja a választ. Az első fázis az emberi tanulást hivatott modellezni, a második fázis pedig a külső segítség nélküli emberi válaszadást. A tübbi módszerek ezzel szemben a kérdések feltétele után kezdik el használni az adatbázisaikat (például Turney módszere az Internetes keresőt). A mi módszerünk ebből a szempontból az LSA-ra hasonlít, mivel a kérdések feltétele előtt mi is egy memória modellt építünk fel (a neuron hálóban szereplő mátrixokat), és a kérdések megismerése után már csak futtatjuk a hálót a bemenet rekonstruálására.

A mi módszerünk egy kicsi, 700 megabájtos adatbázist használ, ami jóval kisebb mind a Turney, mind a Terra által használtnál. Ezt figyelembe véve az elért 76.25% nagyon jónak mondható. Hozzá kell azonban tenni, hogy mi lexikális adatbázist is használtunk, míg Turney, Terra és Landauer módszerei nem. Ki kell viszont hangsúlypzni, hogy a kérdés szó jelentésére vonatkozó statisztikai információkat mindíg töröltük a rendszerből. A lexikális adatbázis csak az ismert szavak jelentéseivel kapcsolatos információkat szolgáltatta számunkra. Fontos továbbá, hogy a WordNet-ben található különféle információkat a szavak és jelentések kapcsolatairól nem használtuk ki, csak a szavak szinoníma halmazokba csoportosítását.

Az adatbázisok szempontjából Jarmasz és Szapakovicz megoldása az eddigiek szöges ellentéte, mivel ők nem használtak szövegadatbázist, csak lexikális adatbázist. A rendszerük nagy mértékben támaszkodik a Roget's Thesaurus nevű gazdag lexikális adatbázisra, amely részletes, szemantikus relációkat leíró hálózatot tartalmaz a nyelv szavaihoz, így sikerült 78.75%-ot elérniük.

Mindent összevetve, az egyéb módszerek vagy nagyobb szövegadatbázist, vagy gazdagabb lexikális adatbázist használtak a miénknél. Tehát a mi módszerünk akkor is hatékony lehet, amikor nincs annyi rendelkezésre álló információ, például, ha korlátozott számú dokumentumban szeretnénk kideríteni egy ismeretlen szó jelentését. Habár használjuk a WordNet-et és a SemCor-t, nem szükséges, hogy tartalmazzák az ismeretlen szót. Ezek csak az ismert

szavak jelentéseire vonatkozó információkat hivatottak szolgáltatni.

Turney kombinált módszerének hatékonysága azt mutatja, hogy az igazán jó teljestmény érdekében a különböző módszerek kombinálása célravezető lehet. A nála leghatékonyabbnak bizonyult szakértők szorzata módszer a mi módszerünket is felhasználhatná egy olyan szakértő rendszerben, mint az övé. Valószínűleg további javítások lennének elérhetőek, ha a felhasználó visszajelzéseit megerősítéses tanulás formájában a rendszerbe integrálnánk.

Számunkra a WordNet szinoníma halmaz rendszere túl finom felbontásúnak bizonyult. A legtöbb szó csak egy jelentésben fordul elő, és nagyon sok szinoníma halmaz csak egyetlen szóból áll. Emiatt egy-egy jelentés nem lesz sokkal jobban definiált, mint egy-egy szó, pedig a rendszer pont ez akarja kihasználni. Viszont a szavakból alkotott kontroll háló eredményi azt mutatták, hogy a szinoníma halmazokat a rendszer így is előnyére tudta használni. Az is egy hasznos tulajdonság, hogy válaszként a felhasználó jelentéseket kap az ismeretlen szó magyarázására, nem pedig szavakat, amik önmagukban még mindíg lehetnek többértelműek. Valószínűleg egy durvább szinoníma halmaz rendszer több információt továbbítana a rendzserünk számára a jelentésekkel kapcsolatban, és ez tovább javítaná az eredményeket.

Az rekonstrukció során negatív aktivitások is kialakulhattak a rejtett reprezentációs rétegekben. Az általunk megadott mátrixok, mint bázisvektorok esetén ez szükséges az optimális rekonstrukcióhoz, viszont ezek az értékek nem értelmezhetőek válaszként. Azt mondjuk, minél nagyobb súllyal szerepel egy bázisvektor a rekonstrukcióban, annál több a közös rész a bemenet, és a bázisvektorhoz tartozó jelentés kontextusában. Viszont ez a gondolatmenet negatív aktivitásokra nem vihető át. A negatív súlyú komponensek szerepe az lehet, hogy kioltsák bizonyos pozitív komponensek hatását, hogy együttesen valami mást legyenek képesek rekonstruálni, mint külön-külön. Az ideális eset az lenne, ha olyan bázisvektoraink lennének, amik pozitív kombinációjával viszonylag jól lehet rekonstruálni mindenféle bemenetet. Vannak olyan háló tanítási stratégiák, amelyek ezt a feltételt tartják szem előtt (NMF), és olyan bázisvektorokat találnak, amelyek pozitív kombinációjaként állíthatók elő a bemenetek közelítései. Egy ilyen tanítási módszerrel valószínűleg jobban értelmezhető rekonstrukciós aktivitások keletkeznének, ami javíthatná a rendszer teljesítményét.

A TD megszorítások nem javítottak, hanem rontottak a teljesítményen, ami szintén azt mutatja, hogy a rendszernek szüksége van kellő számú bázisvektorra a rekonstrukció során. Ezek hiányában a rekonstrukció nem lesz megfelelő, és ez kihat a válasz szavakhoz tartozó jelentések aktivitásainak relatív sorrendjére.

A válasz szavak súlyaira adott különböző küszöbök segítségével növelni tudtuk a módszer találati arányát. Habár ez egy TOEFL teszt megoldása

során nem jelent előnyt, más alkalmazások esetén fontos lehet, hogy ne adjunk rossz választ abban az esetben, ha csekély a bizonyosságunk a válasz helyessége felől.

4. Kitekintés

Mint azt a bevezetőben említettem az intelligens adatbányászat célja olyan keresőrendszerek létrehozása, amelyek hatékonysága jobb a maiakénál, abban az értelemben, hogy olyan dokumentumokat találnak számunkra, amelyek ténylege relevánsak, és ami releváns, azt lehetőleg meg is találják. Tehát, ha lehet, ne adjanak vissza hamis találatokat, és függetlenül attól, hogy milyen szavakkal fogalmazzuk meg a kérdést, találják meg a válaszokat még akkor is, ha egy-egy dokumentum más szóhasználattal írja le azt, amit keresünk.

Jelenlegi munkámban egy ilyen rendszer első verziójának kifejlesztésében veszek részt. A rendszer a mai, jól bevált, kulcsszavas keresőrendszerek egyfajta továbbfejlesztéseként fogható fel, abban az értelemben, hogy egy kulcsszóhalmaz vagy kulcsszavakból felépített kifejezés helyett egy úgynevezett kulcsszóháló játszaná az egyik központi szerepet a keresésben. A keresést intelligens kereső ügynökök, úgynevezett *crawlerek* végeznék, amelyek képesek olyan web site-ok felkutatására, ahol sok, a felhasználó által relevánsnak ítélt dokumentum található [27]. A felhasználó kulcsszavak, és közöttük lévő relációk formájában adhatná meg, hogy mi is az, amit fontosnak talál, mi is a keresése célja.

A kulcsszóháló szavakból felépített szemantikus hálót jelent, melynek első verziójában a szavak közötti élekhez csak egy súly van hozzárendelve. Az élsúlyok a szavak együttes előfordulási frekvenciáiból származtathatók (lásd 3.3.2). Későbbi verzióban a kapcsolatoknak egyéb tulajdonságai is lehetnének, például a kapcsolat megnevezése. Néhány kapcsolatnak kiemelt szerep adható, például ilyen az *is-a*, vagy a *része* reláció. Egy ilyen kulcsszóháló már közel állna a Semantic Web-bel kapcsolatos törekvések körében népszerű ontológiákhoz, amik tulajdonképpen szemantikus hálók, és segítségükkel gazdagon leírhatók a fogalmak közötti kapcsolatok.

Az ügynök intelligens volta abban nyilvánul meg, hogy képes olyan helyek megtalálására az interneten, ahol sok olyan dokumentum van, ami releváns a felhasználó számára. A kereső ügynökök által talált dokumentumokból generált kulcsszóhálót összevetve a felhasználó által megadott kulcsszóhálóval a ügynök eldönti, hogy a dokumentum fontos-e a felhasználó számára. Ha sok ilyet talál egy web site-on, akkor elidőzik ott, ha nem akkor továbbáll abban az irányban, amit a legígéretesebbnek talál. A talált dokumentumok alapján a felhasználó módosíthatja a kulcsszóhálót, például kiegészítheti azt.

Ezen kívül a talált dokumentumokra megerősítést vagy büntetést adhat a rendszernek, így a rendszer képes lehet megerősítéses tanulás segítségével megtanulni azt, hogy mi is az, amit a felhasználó szeretne. Így képes lehet alkalmazkodni az egyes felhasználókhoz, ami igazán rugalmassá tehet egy keresőrendszert.

Hivatkozások

[1] *Altavista*, `http://www.altavista.com`.

[2] *Claws: part-of-speech tagger for english*, `http://www.comp.lancs.ac.uk/computing/research/ucrel/claws/`.

[3] *The darpa agent markup language*, `http://www.daml.org`.

[4] *Extensible markup language (xml)*, `http://www.w3c.org/XML/`.

[5] *Google*, `http://www.google.com`.

[6] *The ontology inference layer (oil)*, `http://www.ontoknowledge.org/oil`.

[7] *Resource description framework (rdf)*, `http://www.w3c.org/rdf`.

[8] *Semantic web*, `http://www.w3c.org/2001/sw/`.

[9] *Test of english as a foreign language (toefl)*, `http://www.ets.org/`.

[10] *Web ontology language (owl)*, `http://www.w3c.org/owl`.

[11] *Wordnet*, `http://www.cogsci.princeton.edu/ wn/`.

[12] Hawoong Jeong Albert-László Barabási, Réka Albert, *Scale-free characteristics of random networks: the topology of the world-wide web*, Physica A **281** (2000), 69–77.

[13] Pedro Domingos AnHai Doanm Jayant Madhavan and Alon Halevy, *Learning to map between ontologies in the semantic web*, Tech. report, Computer Science and Engineering, University of Washington, Seattle, WA, USA, 2002.

[14] D. Widdows B. Dorow and K. Ling, *Using curvature and markov clustering graphs to for lexical acquisition and word sense discrimination*, MEANING-2005, 2nd Workshop organized by the MEANING Project (Trento, Italy), February 3rd-4th 2005.

[15] D. Beate and W. Dominic, *Discovering corpus-specific word senses*, EACL 2003, Budapest, 2003, pp. 79–82.

[16] Rebecca Bruce and Janyce Wiebe, *Word sense disambiguation using decomposable models*, 32nd Annual Meeting of the Associations for Computational Linguistics (Las Cruces, New Mexiko), 1994, pp. 139–145.

[17] Rudi Cilibrasi and Paul Vitanyi, *Automatic meaning discovery using google*, arXiv:cs.CL/0412098 v1, December 21 2004.

[18] Ellen M. Voorhes Claudia Leacock, Geoffrey Towell, *Towards building contextual representations of word senses using statistical models*, Corpus Processing for Lexical Acquisition, 1996, pp. 97–113.

[19] P. Domingos and M. Pazzani, *On the optimality of the simple bayesian classifier under zero-one loss*, Machine Learning **29** (1997), no. 2/3, 103–130.

[20] I. H. Witten C. Gutwin E. Frank, G. W. Paynter and C. G. Nevill-Manning, *Domain-specific keyphrase extraction*, Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99) (California: Morgan Kaufmann), pp. 668–673.

[21] P. Edmonds, *Choosing the wordmost typical in context using a lexical cooccurrence network*, In Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics, Madrid, 1997, pp. 507–509.

[22] Philip A. Bernstein Erhard Rahm, *A survey of approaches to automatic schema matching*, The VLDB Journal, vol. 10, 2001, pp. 334–350.

[23] C. Fellbaum, *Wordnet: An electronic lexical database.*, MIT Press, Cambridge, MA, 1998.

[24] R. Garside G. Leech and M. Bryant, *Claws4: The tagging of the british national corpus*, In Proceedings of the 15th International Conference on Computational Linguistics (COLING 94), 1994, pp. 622–628.

[25] Marti A. Hearst, *Noun homograph dismbiguation using local context in large text corpora*, 7th Annual Conference of the University of Waterloo Centre for the New OED and Text Research, 1991.

[26] Ramon Ferrer i Cancho and Richard V. Solé, *The small world of human language*, Royal Society, vol. 268, 2001, pp. 2261–2265.

[27] András Lőrincz István Kókai, *Fast adapting value estimation-based hybrid architecture for searching the world-wide web*, Applied Soft Computing **28** (2002), 1–13.

[28] M. Jarmasz and S. Szpakowicz S., *Roget's thesaurus and semantic similarity*, In Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP-03), 2003.

[29] Holger Ebel Jörn Davidsen and Stefan Bornholdt, *Emergence of small world from local interactions: Modeling acquaintance networks*, Physical Review Letters **88** (2002), no. 12.

[30] Jon Kleinberg, *The small world phenomenon: An algorithmic perspective*, Tech. Report 99-1776, Department of Computer Science, Cornell University, Ithaca NY 14853, October 1999.

[31] B. Krulwich and C. Burkey, *Learning user information interests through the extraction of semantically significant phrases*, AAAI 1996 Spring Symposium on Machine learning in Information Access (M. Hearst and H. Hirsh, eds.), 1996.

[32] T. K. Landauer and S. T. Dumais, *A solution to plato's problem: Thelatent semantic analysis theory of acquisition, induction andrepresentation of knowledge*, Psychological Review **104** (1997), 211–240.

[33] Dekang Lin, *Using syntactic dependency as local context to resolve word sense ambiguity*, Tech. report, Department of Computer Science, University of Manitoba, Winnipeg, Manitoba, Canada, R3T 2N2, 1998.

[34] Dekang Lin and Randy Goebel, *Context free grammar parsing by message parsing*, Tech. report, Department of Computer Science, University of Manitoba, Winnipeg, Manitoba, Canada, R3T 2N2, 1995.

[35] C. D. Manning and H. Schütze, *Foundations of statistical natural language processing*, MIT Press, Cambridge, MA, 1999.

[36] Josh Tenenbaum Mark Steyvers, *The large-scale structure of semantic networks*, Tech. report, Department of Psychology, Stanford University, Stanford, CA 94305-2130, 2000.

[37] S. Milgram, *The small world problem*, Psychology Today **2** (1967), 60ü67.

[38] A. Munoz, *Compund key word generation from document databases using a hierarchical clustering art model*, Intelligent data analysis (Amsterdam: Elsevier), 1996.

[39] Jean Véronis Nancy Ide, *Word sense disambiguation: The state of the art*, Computational Linguistics **24** (1998), no. 1.

[40] Borys Omelayenko, *Learning of ontologies for the web: the analysis of existent approaches*, Proceedings of the International Workshop on Web Dynamics, held in conj. with the 8th International Conference on Database Theory (ICDT01) (London, UK), 3 January 2001.

[41] Patrick Pantel and Dekang Lin, *Discovering word senses from text*, SIGKDD (Edmonton, Alberta, Canada), July 23-26 2002, pp. 613–619.

[42] Patrick Pantel and Deepak Ravichandran, *Automatically labelling sematic classes*, Tech. report, Information Science Institute, University os Southern California, 4676 Admiralty Way, Marina del Rey, 90292, 2003.

[43] J. Bigham Peter D. Turney, M. L. Littman and V. Shnayder, *Combining independent modules to solve multiple-choice synonym and analogy problems*, Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP-03), Borovets, Bulgaria, 2003, pp. 482–489.

[44] Gio Wiederhold Prasenjit Mitra and Stefan Decker, *A scalable framework for the interoperation of information sources*, Tech. report, Infolab, Stanford University, Stanford, CA, USA 94305, 2002.

[45] Jan Jannink Prasenjit Mitra, Gio Wiederhold, *Semi-automatic integration of knowledge sources*, Tech. report, Infolab, Stanford University, Stanford, CA, USA 94305, 1998.

[46] Erzsébet Ravasz and Albert-László Barabási, *Hierarchical organization in complex networks*, arXiv:cond-mat/0206130 v2, September 3 2002.

[47] Duncan J. Watts & Steven H. Strogatz, *Collective dynamics of 'small-world' networks*, Nature **393** (1998), 440–442.

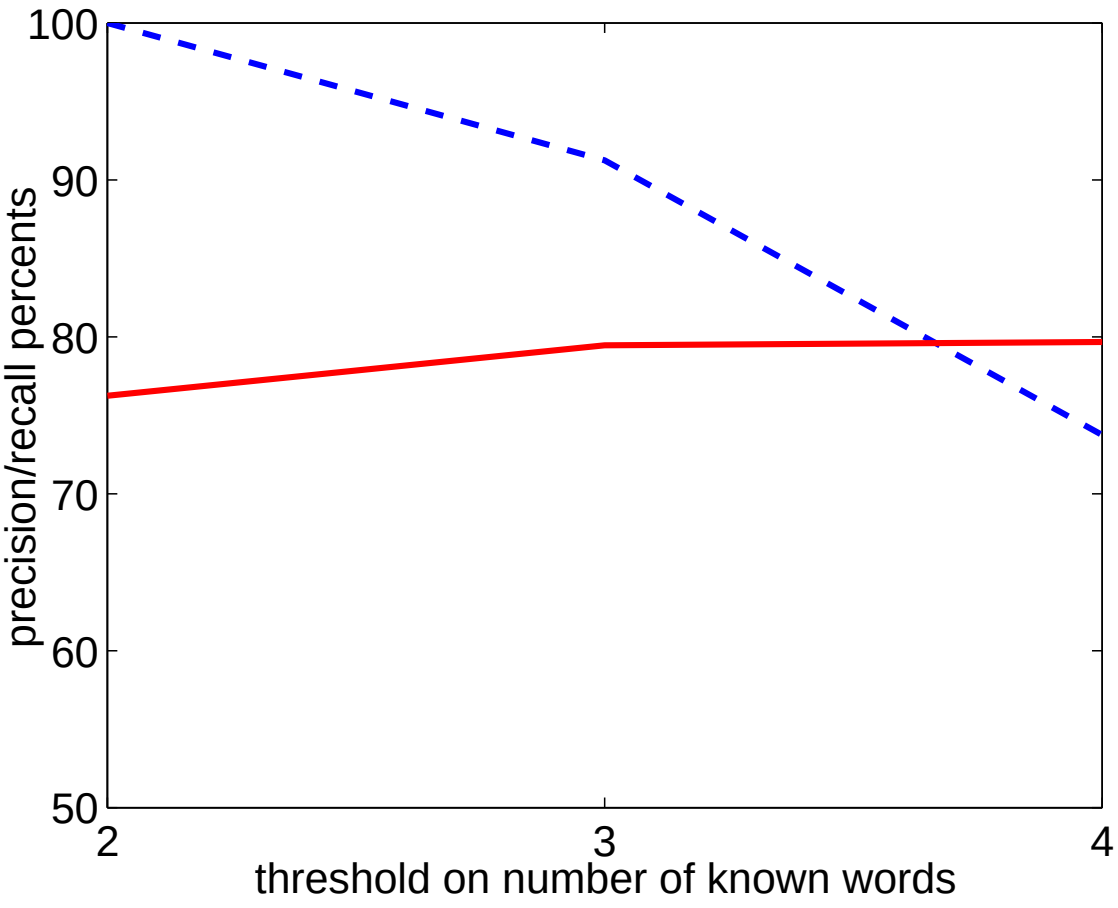
[48] E. Terra and C. L. A. Clarke, *Choosing the wordmost typical in context using a lexical co-occurrence network*, In Proceedings of the Human Language Technology and North American Chapter of Association of Computational Linguistics Conference, 2003, pp. 244–251.

[49] Peter D. Turney, *Learning algorithms for keyphrase extraction*, Information Retrieval (1999).

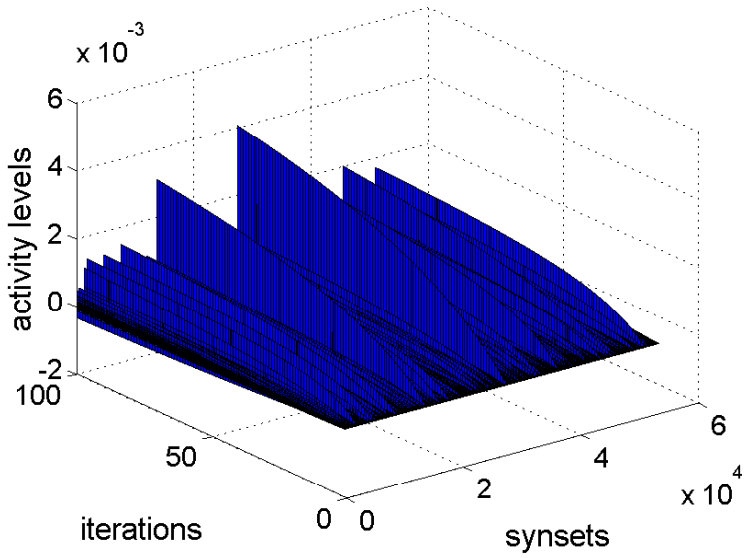
[50] ———, *Mining the web for synonyms: Pmi-ir versus lsa on toefl*, Proceedings of the Twelfth European Conference on Machine Learning (ECML-2001), Freiburg, Germany, 2001, pp. 491–502.

[51] ———, *Coherent keyphrase extraction via web mining*, Tech. report, Institute for Information Technology, National Research Council of Canada, M-50 Montreal Road, Ottawa, Ontario, Canada, K1A 0R6, 2002.

[52] Stijn Marinus van Dongen, *Graph clustering by flow simulation*, Ph.D. thesis, University of Utrecht, 2000.



3. ábra. **A találati arány és a válaszadási arány változása:** A folytonos vonal a találati arányt, a szaggatott vonal a válaszadási arány változását jelöli. Az (a) ábra az ismert szavak számára adott küszöbölés esetét mutatja, a (b) ábra az első és a második súly arányára alkalmazott küszöbölés esetét, míg a (c) ábrán az első súlyra alkalmazott küszöbölés eredménye látható.



4. ábra. **A rejtett réteg aktivitásainak változása az iteráció során**