Egy hatékonyabb webes sablonszűrő algoritmus – avagy miként lehet a cumisüveg potenciális veszélyforrás Obamára nézve

Endrédy István[1], Novák Attila[1]

[1] MTA–PPKE Nyelvtechnológiai Kutatócsoport
1083 Budapest, Práter utca 50/a
{endredy.istvan.gergely, novak.attila}@itk.ppke.hu

Kivonat: A folyamatosan növekvő web[1] tartalmából hatékonyan lehet nagyméretű korpuszt építeni. Azonban a dinamikusan generált weblapok gyakran sok irreleváns és ismétlődő szöveget tartalmaznak, amelyek egyes sablonos szövegrészeket, kifejezéseket felülreprezentálva rontják a korpusz minőségét. Ebben a cikkben olyan automatikus szövegkinyerő eljárást mutatunk be, amely a korábbi módszereknél hatékonyabban minimalizálja az irreleváns ismétlődő részek előfordulását a webről letöltött korpuszokban.

1 A feladat

A webről nyert korpuszok építésekor az általában dinamikusan generált HTML-tartalomból történő szövegkinyerés nem triviális feladat a különböző oldalakon ismétlődő rengeteg irreleváns sablonos tartalom miatt. A feladatot az angol irodalomban boilerplate removalnek, sabloneltávolításnak nevezik. A művelet lényege, hogy a HTML tartalomból csak az értékes részeket igyekszik kiszűrni, a menük, fej- és láblécek, reklámok, a minden oldalon ismétlődő struktúra kiszűrésével.

A fenti feladatra számos algoritmus létezik. Ezek közül két szabadon hozzáférhető implementációval rendelkező megoldást alaposabban is szemügyre vettünk.

1.1 A Body Text Extraction (BTE) algoritmus

A BTE [1] a weblap azon általában jellemző tulajdonságát használja ki, hogy a boilerplate tartalomban sűrűbben szerepelnek a HTML-tagek. Ezért megkeresi azt a leghosszabb részt, ahol a tagek száma minimális. Az ötlet egyszerű, azonban sokszor téved, és nem képes szöveget kinyerni olyan helyzetekben, ahol az értékes részben a feltételezéssel ellentétben mégis sűrűbben fordulnak elő HTML-tagek, például ha táblázat szerepel benne, vagy egy reklámokkal szabdalt cikk esetében. Ilyenkor az értékes tartalom jelentős része (akár az egész) elveszhet, esetleg helyette teljes egészében irreleváns tartalom jelenik meg.

[1] 50 milliárd oldal, http://www.worldwidewebsize.com (2012.09.20.)

1.2 JusText algoritmus

A JusText algoritmus [2] a HTML tartalmat bekezdésekre bontja a szöveget tartalmazó tagek mentén. Minden egységben megszámolja a benne szereplő linkek, stopwordök és szavak számát. Ezek alapján osztályozza őket: bizonyos küszöb mentén *jó*, *majdnem jó* illetve *rossz* kategóriákba. Majd a *jókkal* körülvett *majdnem jók* szintén bekerülnek a *jók* közé, és így születik meg az értékes szöveg: a *jó* bekezdések. Az algoritmus elég jó minőséget ad még extrém oldalakon is.

Az algoritmus használatával 2012 nyarán magyar hírportálokról kinyert tartalmat vizsgálva azonban feltűnt, hogy a kapott korpusz még mindig a várthoz képest nagyon erősen túlreprezentált kifejezéseket tartalmaz, mint az alábbi példák mutatják:

(1) Utasi Árpi-szerű mesemondó . (10587 db)
cumisüveg potenciális veszélyforrás . (1578 db)
Obama amerikai elnök , (292 db)

(2) etióp atléta: cseh jobbhátvéd (39328 db)
Barack Obama amerikai elnök (2372 db)
Matolcsy György nemzetgazdasági miniszter (1633 db)
George Bush amerikai elnök (1626 db)

Azt találtuk, hogy a problémát elsősorban egyrészt a cikkek alján található (kapcsolódó) cikkajánlók nem megfelelő kiszűrése, másrészt a kizárólag ilyeneket tartalmazó oldalak okozzák.

2 Az aranyásó algoritmus

A problémát az adott egyedi weboldalaknál magasabb szintre lépve sikerült a korábbinál hatékonyabban megoldani. Eleinte abba az irányba indultunk, hogy az egyes weblapok nem kívánt részeit távolítsuk el. Mikor nem sikerült átütő eredményt elérni, akkor jutott eszünkbe, hogy miért a rosszat keressük, miért nem az értékes részeket? Ahogy az életben is érdemes a jót keresni, így tettünk a weboldalak esetén is: így született meg az aranyásó algoritmus.

A megoldás azon alapul, hogy egy adott webdoménen/aldoménen belül a dinamikusan generált weboldalak, illetve url-ek nem szöveges tartalma (pl. a HTML-kód) általában tartalmaz közös mintázatokat, amelyeket azonosítva megtalálható az értékes tartalom. Az algoritmus az adott domén oldalaiból mintát vesz, és megpróbálja megkeresni azt a HTML-taget, amelyen belül (az oldalak zömében) a cikk található, különös tekintettel azokra a mintákra, amelyek az oldalakon ismétlődnek. Például gyakori a hírportálokon, hogy a cikk alján feltüntetik a legnépszerűbb 5 cikk ajánlóját. Ezeket a szövegeket a korábbi sablonszűrő algoritmusok nem szűrik ki (pl. a justext a cikk részének veszi őket), mert önmagukban nem rossz szövegek, viszont duplikátumot okoznak majd a korpuszban, erősen felülreprezentálva az ezekben található szöveget.

Minden doménre megtanuljuk azt a HTML szülő taget, amely csak a cikket tartalmazza, majd csak ezen tag tartalmát adjuk oda a sablonszűrő algoritmusnak. Előnye, hogy azon oldalak, ahol nincs cikk (tematikus nyitólapok, címkefelhők, keresőlap-eredmények, stb.), ott az algoritmus nem ad semmit, hiszen a doménre jellemző cikk tag nincs jelen. Így az algoritmus automatikusan kiszűri ezeknek a lapoknak a tartalmát, ami örvendetes. Azokon a lapokon pedig, ahol valóban van cikk, a sablonszűrő algoritmus már csak a lényegi tartalmat kapja, erősen megkönnyítve a dolgát.

2.1 Az algoritmus részletes leírása

Első lépésként megtanuljuk a doménre jellemző HTML-mintázatot, ez a tanulófázis. Az algoritmus letölt pár 100 oldalt, és lefuttatja rajtuk a sablonszűrő algoritmusokat (justext), amely minden oldal tartalmát bekezdésekre bontja és értékeli. Az Aranyásó algoritmus az egyes kinyert bekezdéseket átnézi: ismétlődnek-e más oldalakon a letöltött mintában. Ha igen, akkor ezeket rossz bekezdésnek minősíti, majd megkeresi a jó bekezdések legközelebbi, közös szülő tagjét a DOM hierarchiában. A tanulófázis végén a leggyakoribb jó szülő tag lesz a győztes. Nem kapunk optimális eredményt azonban, ha végpontként ennek a tagnek a zárópontját választjuk. Előfordul ugyanis, hogy a teljes cikket tartalmazó szülő tag nem kívánt bekezdéseket is tartalmaz. Ilyen esetben az algoritmus nem találná meg az optimális vágási pontokat. Ezért az első körben választott tartalmon belül újabb keresést végzünk az optimális végpontra, amely több tagből álló sorozat is lehet. Miután ezt is kiválasztottuk, az algoritmus (természetesen a tanulófázis oldalaiból is) csak az így kivágott rész tartalmát adja át a sablonszűrőnek.

Az Aranyásó csak azokból az oldalakból tanul, amelyben a kinyert bekezdések hossza elér egy minimumot: pár doménen, ahol rendkívül gyakoriak a tényleges tartalmat nem adó gyűjtőlapok, e nélkül nem sikerült a megtanulni a legjobb vágási pontot.

3 Eredmények

Az Aranyásó algoritmussal jelentősen sikerült csökkentenünk a legyűjtött korpuszban szereplő ismétlődéseket. Három doménen (origo.hu, index.hu, nol.hu) való futtatással 2000 oldal letöltése után a bejárást leállítva kapott eredményeinket a következő oldalon szereplő táblázatban foglaljuk össze.

1. táblázat: Sablonszűrő algoritmusok összehasonlítása az egyes doméneken.

Algoritmus		Kinyert mondatok száma	Egyedi mondatok száma	Össz karakter-szám	Egyedi monda-tok karakter-száma	Egyedi mondatok aránya %	Egyedi mondatok aránya karakter-számban %
origo	összes szöveg	264 423	63 594	16 218 753	7 048 011	24%	43%
	BTE	60 682	33 269	12 016 560	7 499 307	54%	62%
	JusText	58 670	30 168	8 425 059	4 901 528	51%	58%
	aranyásó	22 475	21 242	3 076 288	3 051 376	94%	99%
nol.hu	összes szöveg	509 408	144 003	25 358 477	12 570 527	28%	49%
	BTE	154 547	107 573	24 292 755	13 544 130	69%	55%
	JusText	186 727	128 782	14 167 718	11 665 284	68%	82%
	aranyásó	162 674	123 716	12 326 113	11 078 914	76%	89%
index.hu	összes szöveg	232 132	55 466	9 115 415	4 542 925	23%	49%
	BTE	51 713	26 176	5 756 176	4 061 697	50%	70%
	JusText	40 970	29 223	4 371 693	3 441 337	71%	78%
	aranyásó	13 062	11 887	1 533 957	1 489 131	91%	97%

Az eredmények világosan mutatják, hogy az algoritmus hatékonyan csökkenti a kinyert korpuszban szereplő felesleges ismétlődéseket. Arra vonatkozó becslést egyelőre nem végeztünk, hogy ugyanakkor mennyi értékes anyagot veszítünk esetleg el, és hogy ebből a szempontból az eredmény hogyan viszonyul az egyéb algoritmusok teljesítményéhez.

Nézzük tehát, hogy a módosított algoritmust novemberben futtatva hogyan teljesít Obama elnök a cumisüveggel szemben (hasonlóan az (1) példához, a kifejezés után álló írásjel itt is a minta részét képezi):

(3) Obama amerikai elnök , (47 db)
Utasi Árpi-szerű mesemondó . (1 db)
cumisüveg potenciális veszélyforrás . (1 db)

Látjuk, hogy az eredeti mintában a vesszőt is tartalmazó '*Obama amerikai elnök ,*' kifejezés maga is felül volt reprezentálva, nem is beszélve a másik kettőről. Mi történt ugyanakkor az etióp atlétával és a cseh jobbhátvéddel?

(4) Matolcsy György nemzetgazdasági miniszter (694 db)
Barack Obama amerikai elnök (664 db)
Sólyom László köztársasági elnök (367 db)

Angela Merkel német kancellár (345 db)
...
etióp atléta: cseh jobbhátvéd (1 db)

Eltűntek ők is ki kicsoda-listánkról. Bár a teljes domént az algoritmus a cikk írásakor még nem járta be, ezért adataink nem teljesen összehasonlíthatók a korábbiakkal, megnyugtató, hogy a gyanús kifejezések eltűntek a gyakori kifejezések listájáról.

Hivatkozások

1. Finn, A., Kushmerick, N., Smyth, B.: Fact or fiction: Content classification for digital libraries. In: DELOS Workshop: Personalisation and Recommender, Systems in Digital Libraries (2001)
2. Pomikalek, J.: Removing Boilerplate and Duplicate Content from Web Corpora. Masaryk University Faculty of Informatics, Brno, 2011. http://code.google.com/p/justext/ (2012. május 20.)