Közös crawlnak is egy korpusz a vége – Korpuszépítés a CommonCrawl .hu domainjából

Indig Balázs

Pázmány Péter Katolikus Egyetem, Információs Technológiai és Bionikai Kar
MTA-PPKE Magyar Nyelvtechnológiai Kutatócsoport
{indig.balazs}@itk.ppke.hu

Kivonat Napjainkban az interneten fellelhető szövegek mennyisége és könnyű elérhetősége miatt a nagy korpuszok előállítása egyre könnyebb és szabványosabb. A kihívást manapság az internetes oldalak változékonysága jelenti. Bizonyos tartalmak eltűnnek és újak keletkeznek, valamint az oldalak közötti kapcsolatok is folyamatosan változnak. Egyre népszerűbbek az olyan közhasznú kezdeményezések, melyek „az internetet akarják archiválni". Az ilyen szolgáltatások a leszedett tartalmakat ingyen elérhetővé teszik bárki számára. A cikkben az így elérhetővé tett tartalmak minőségi jellemzőit vizsgáljuk korpusz-előállítás szempontjából, különös tekintettel az Amazonon tárolt Common Crawl archívumból elérhető magyar nyelvű tartalmakra. Célunk egy reprodukálható, könnyen elérhető technikákkal előállított magyar nyelvű webkorpusz létrehozása, mely publikusan elérhető. A korpusz így a nemrégiben elérhetővé vált `e-magyar` digitális nyelvfeldolgozó rendszer kiváló nyersanyagát fogja képezni.
Kulcsszavak: web crawler, korpuszépítés

1. Bevezetés

Az interneten megjelenő tartalmak számszerű növekedésével és változásuk gyorsulásával együtt megjelent az az igény, hogy a jövő számára lenyomatot készítsünk és egyfajta „digitális levéltárban" megőrizzük az egyes weboldalak különböző verzióit. Az internet rohamos fejlődése miatt a technológiák gyorsan váltják egymást, így bizonyos, egy időben valamilyen szempontból fontosnak tartott tartalmak gyakran megváltoznak, majd eltűnnek.

Célszerűnek tűnik tehát egyfajta archiválási technika bevezetése. A Wikipedia például a kezdetektől fogva „vandálbiztos", azaz minden változtatás egy új verziót hoz létre az adott oldalból és a változások visszakövethetők, ahogy az napjainkra a programok forráskódjánál alapvető elvárás. Ez az ismérv jelenleg még csak nagyon kevés oldalra jellemző. Ezért több elképzelés született az internetes oldalak külső fél általi archiválására, melyek napjainkra szabványosítva lettek. Ebben a formátumban több szervezet is közzé teszi az általa gyűjtött tartalmakat.

A nyelvtechnológia célja természetesen elsődlegesen nem az archiválás, hanem az archivált anyagok elemzése és feldolgozása. A cikkben az *Amazon* által

tárolt és szolgáltatott Common Crawl archívum példáján mutatjuk be, hogy hogyan használható egy ilyen archívum arra, hogy reprodukálható módon korpuszt lehessen belőle építeni, mely később az eszközök fejlődésével újra és újra, jobb minőségben születhet meg. A létrejött szabadon elérhető korpusz kiváló kiegészítője a nemrégiben publikált szabadon elérhető teljeskürű eszközláncnak, az `e-magyar` digitális nyelvfeldolgozó rendszernek [1].

A cikkben először összehasonlításképpen bemutatjuk az elérhető magyar nyelvű korpuszokat, majd a web archiválásának fontosabb szereplőit és szabványait. Ezek után a létrehozandó korpuszhoz szükséges módszert ismertetjük, a cikk végén pedig a létrehozott korpusz és a Common Crawl archívumban található magyar nyelvű tartalmak jellemzőit tárgyaljuk.

2. Előzmények

2.1. Fontosabb magyar nyelvű korpuszok

Magyar nyelvre több nagyobb korpusz is létezik, melyek részben vagy egészben a webről származnak (lásd az 1. táblázat). A bárki számára feldolgozott állapotban azonnal hozzáférhető *Webkorpusz* [2] szűretlen változata 1,48 milliárd, míg a szűrt változata 589 millió szót tartalmaz. A maga idejében a legnagyobb magyar nyelvű korpusz volt, mely közvetlenül elérhető egy rendkívül megengedő licenc alatt. A korpusz forrásául a .hu domain 2003-as állapota, azaz 18 millió letöltött oldal szolgált. A gyűjtött anyag jól reprezentálja az írott nyelv igen széles skáláját. A készítés során a többször is megtalálható szövegek kiszűrésre kerültek, ahogy azok a fájlok is, amelyek nem tartalmaztak használható szöveget. A szűrés több lépcsőben történt, aszerint, hogy egy oldalon belül a szavak hány százalékát ismerte fel a helyesírás-ellenőrző program. A szerzők által is jó minőségűnek tartott, és közzétett korpuszrészlet a teljes korpusz 4%-át tartalmazza.

A Webkorpusz készítése során a letöltött oldalak teljes HTML változata nem lett megőrizve a nagy tárhelyigény miatt. Viszont az elkészítéséhez használt eszközök minősége sokat javult azóta, melyekkel az eredeti HTML oldalak hiányában nem reprodukálható a letöltött nyers oldalak feldolgozásának folyamata. Ez a tény a korpusz minőségén történő utólagos javítást rendkívüli módon megnehezíti.

A *Magyar Nemzeti Szövegtár (MNSZ)* [3,4] két fő verzióját különböztetjük meg. Az első változata 187 millió szövegszót tartalmaz, míg a második kiadás (MNSZ 2) az elsőt magába foglalva 1,348 milliárd szóból áll[1]. A korpusz különlegessége, hogy nem csak webről származó szövegeket tartalmaz, hanem van egy határon túli magyar nyelvű szövegeket valamint egy beszélt nyelvi átiratokat tartalmazó alkorpusza is.

A nyers szövegek, illetve a feldolgozáshoz szükséges eszközök nem szabadon letölthetőek egy eszközláncként, ahogy maga a korpusz is csak kérésre elérhető

[1] A korpuszon nem történt duplikációszűrés, így tényleges méret ennél valamivel kisebb lehet az ismétlődések miatt.

szöveges formában, valamint regisztráció után a keresőfelületen használható. A reprodukció és a minőség javítása ezért egyedül a korpusz készítőitől függ.

A *Pázmány korpusz* [5] teljes egészében a 2015-ös magyar webről származó szövegeket tartalmaz. Az 1,24 milliárd token célzottan kiválasztott magyar hírportálokról származó oldalakon található, duplikációszűrt szövegekből áll. A korpuszban meg vannak különböztetve továbbá a hírportálok szerkesztett, a hírhez tartozó szövegrészei, valamint a szerkesztetlen, a hírhez érkezett kommentek szövegeit tartalmazó alkorpuszok.

A készítésekor a szerzők az általuk kitalált speciális *Goldminer-algoritmust* [6] használták, mely *crawl*-olás közben egy mintavétel alapján egy oldal kinézetét jobban meghatározva hatékonyabb sablonszűrést (*boilerplate removal*) tesz lehetővé, mint az addigi state-of-the-art megoldás, ami külön-külön vizsgálta az oldalakat. Sajnos a korpusz jelenleg nem érhető el szabadon. A készítéshez használt eszközök egy része elérhető[2], de nem közvetlenül felhasználható formában. A nyers HTML oldalak ebben az esetben sem lettek megtartva a magas tárhelyigény miatt.

Név	Méret (token)	forrás	készítés éve
Webkorpusz (teljes)	1,48 milliárd	web vegyes	2003
Webkorpusz (4%)	589 millió	web vegyes	2003
Magyar Nemzeti Szövegtár I	187 millió	újság, beszédátirat, web vegyes	2001
Magyar Nemzeti Szövegtár II	1,348 milliárd	újság, beszédátirat, web vegyes	2017
Pázmány korpusz	1,24 milliárd	web hírportálok, kommentek	2015

1. táblázat. A főbb magyar nyelvű korpuszok adatai

Látható, hogy a korpuszépítés során problémát jelent a nyers HTML oldalak tárolása, illetve az esetek jó részében a korpuszt készítő programok sem elérhetőek teljes mértékben. A következő fejezetben az előbbi probléma egy lehetséges megoldását ismertetjük.

2.2. Web archiválása napjainkig

Az internet gyorsan változó világában nagyon fontos a pontos metainformációkkal együtt történő precíz és gyors archiválás, ugyanis az egyes tartalmak elérhetősége néha igen rövid időt ölel csak fel. A verziókezelés, a duplikátumok szűrése és a keresés is fontos szempont a későbbi feldolgozás céljából. A feladatot több nagyobb szervezet iparági összefogással próbálja megoldani.

Az egyikük, az *Internet Archive* egy San Franciscó-i nonprofit digitális könyvtár, amely szabad hozzáférést biztosít számos digitalizált tartalomhoz, melyek közé tartoznak többek között az archivált weboldalak, valamint szabad jogállású könyvek digitalizált változatai is. A teljes archívum mérete 2016 októberében elérte a 15 petabájtot.

[2] `https://github.com/endredy/GoldMiner`

Bár szabadon is feltölthetőek tartalmak, az anyagok nagy része mégis internetes crawlerek segítségével kerül a rendszerbe. A webarchívumuk, a *Wayback Machine*[3] már több mint 308 milliárd oldalt rögzített a metaadataival együtt.

A szervezet által archiválásra kifejlesztett fájlformátum a *WARC (Web ARChive)* – mely az Internet Archive ARC fájlformátumának (ARC) általánosított, szabványosított változata – meghatározza a pontos módját annak, hogy hogyan kell egységesen elraktározni a sokféle digitális erőforrást a hozzájuk tartozó összes metainformációval együtt. Ezen felül támogatja a keresést és az egyes oldalak közötti navigációt, valamint a duplikátumok hatékony tárolására vonatkozó verzióinformációkat. Manapság ez a formátum a web archiválásának szabványos módjává vált.

A *Common Crawl* egy nonprofit szervezet, amely a saját maga által webről leszedett és archivált tartalmakat nyers formában szabadon hozzáférhetővé teszi a lehető legszélesebb közönség számára. Céljuk, hogy az internetet „tehermentesítsék" azáltal, hogy nem kell mindenkinek magának letöltenie a weboldalakat. 2017 novemberére a webarchívumuk mérete 260 TB-ra rúg, amely 3,2 milliárd weboldalról származik[4]. Az archívumot 2012 óta az Amazon szolgálja ki a publikus adathalmazok programjának részeként[5].

Az eszközök tekintetében 2013-tól kezdve a saját crawler helyett az *Apache*-féle szabadon elérhető *Nutch webcrawler*-t használják a saját implementáció helyett, valamint átálltak a szabványos WARC formátumra. 2015-től kezdve az összes archivált tartalom indexelve is elérhető[6], így könnyebben kereshető az URL-ek szerint is.

A CommonCrawl három formátumban teszi elérhetővé a leszedett tartalmat:

- WARC: a leszedett nyers adatok
- WAT: a WARC formátumban található adathoz tartozó származtatott metadatok
- WET: a WARC fájlokból kinyert szöveges tartalom (a kinyerés módjáról nincs elérhető információ)

A későbbiekben a WARC formátummal foglalkozunk, mivel az számunka elég információt tartalmaz: (1) a *crawl*-olási folyamat pontos leképezését adja, (2) tárolja a HTTP kérést és választ a fejléccel együtt, (3) valamint a tényleges tartalom metaadatai is rögzítve vannak.

3. A magyar web a Common Crawl tükrében

Az látható, hogy a webről történő korpuszépítés módját nagyban meghatározzák az eddig döntően angol nyelvű törekvések. Szerencsére ezek egy az egyben átültethetők magyar nyelvre a szabadon elérhető gazdag eszköztárat felhasználva.

[3] `http://archive.org/web/`
[4] `http://commoncrawl.org/2017/11/november-2017-crawl-archive-now-available/`
[5] `https://aws.amazon.com/public-datasets/common-crawl/`
[6] http://index.commoncrawl.org/

A magyar szövegek kinyerésének szempontjából két problémát fedezhetünk fel. Egyfelől, a szövegkinyerő eljárások terén még számíthatunk fejlődésre, így a leszedett tartalmat meg kell őrizni. Másfelől pedig, a futtatott crawlerek céltalanul bolyonganak a weben, és így az egyes oldalak fontosságuktól függetlenül vannak reprezentálva az archívumban. Ezért célszerű lehet a jövőben azonos technológiával saját crawlereket futtatni, speciálisan a magyar web archiválásának céljából, valamint mindig a state-of-the-art sablonszűrő algoritmus használatával kinyerni a szöveget a leszedett tartalomból.

Jelen cikkben a célunk, hogy megvizsgáljuk, hogy a szabadon hozzáférhető anyagból milyen mennyiségű és minőségű magyar nyelvű korpusz gyártható. Az így készült korpusz előnye az eddigiekhez képest – a méretétől és tartalmától függetlenül – az, hogy az eredmény reprodukálható és szabadon hozzáférhető lesz, valamint folyamatosan frissül és bővül, lehetővé téve diakron elemzéseket is.

3.1. Az architektúra

Az archívum az Amazon által biztosított API-n keresztül hozzáférhető. A fellelhető technikai leírásokból látható, hogy az Amazon legfőbb prioritása, hogy a saját felhős rendszerein használható legyen az adathalmaz, így az elérhető példaprogramokból a felhőn kívüli használat hiányzik. Az elérhető példakódok többnyire Python2 és JAVA nyelven íródtak, de más nyelvekre is elérhetőek, mivel az API a szabványos HTTPS protokollra épül. A cikkben implementált eljárásokhoz a Python nyelvet használjuk, ezért a későbbiekben csak a Python nyelvű API-val foglalkozunk. Bár a példaprogramok csak Python2 nyelven érhetőek el, a szükséges könyvtárak Python3 nyelven is elérhetőek [7], így lehetővé téve a példaprogramok egyszerű átírását, amivel a Python nyelv 3-as dialektusának legújabb verziói adta előnyöket is tudjuk használni.

Az archívum indexe külön szerveren található, az Amazontól függetlenül. Egyszerű REST API kérésekkel hívható és JSON formátumban adja vissza a kívánt kimenetet. A keresett URL-eket tartalmazó WARC fájlok letöltéshez szükséges mezők nevei viszont nehezen találhatók meg a dokumentációban, és a példa programok nem tartalmazzák a lehetséges mezőket. Az API dokumentációja szerencsére kellően részletes a szűrési lehetőségek és az elérhető mezők tekintetében. Maga az archívum havonta frissül, így az index is havi bontásban érhető el. Az indexben URL szerint és a *crawl*-olás hónapja szerint lehet keresni. Maga az index alatt lévő szoftver nyílt forráskódú és szabadon felhasználható saját, azonos formátumban lévő archívumok indexelésére is.

A kívánt oldal letöltéséhez szükséges egy az indexből kiszedett speciális mezőkből összerakott URL, melyet az Amazon sajátos konvencióinak megfelelően kell átadni. Ebben segít az SDK-ban található magas szintű API, melyben a következő lépéseket kell végrehajtani a megfelelő oldal gzip formátumban történő letöltéséhez:

1. Létrehozunk egy munkamenetet az *s3* szolgáltatásra.

[7] `http://boto3.readthedocs.io/en/latest/`

2. Explicit módon az anonim bejelentkezést választjuk.
3. Kiválasztjuk a *commoncrawl* nevű *bucketet*.
4. Megadjuk a *key*-ként az indexből kapott fájlnevet.
5. Megadjuk a *range* paraméterként (szöveges formátumban kötőjellel elválasztva) az indexből kapott *offset* és *length* értékeket.
6. Elküldjük a HTTP kérést, melynek eredményeként megkapjuk a WARC fájlt.

Az így kapott fájl tartalmazza a HTTP kérésre és válaszra vonatkozó adatokat is a tartalom mellett. A metaadattól ezek után egyszerűen különválasztható a tényleges tartalom a WARC fájlformátum tervezéséből adódóan. Ezek után a megkapott oldalból ki kell nyerni a szöveges adatot, és fel kell dolgozni az így kapott nyers szöveget a szokásos szerelőszalagon.

Elmondhatjuk tehát, hogy az architektúra három részből áll, az indexből ki kell nyerni a kívánt oldalak letöltéshez szükséges adatait, le kell tölteni az egyes oldalakat, és a tárhelykapacitás korlátossága miatt röptében feldolgozni őket az ismert eszközökkel. A szabadon elérhető, Python nyelven jól használható eszközök hiánya miatt, csak a *JusTextet* [7] alkalmaztuk demonstrációs céllal, mivel így a letöltött és feldolgozott anyag tárolása nem ütközött korlátokba. A *JusText* program sajátosságából fakadóan elvégezte a szöveg bekezdésekre tagolását is, amelyet megtartottunk.

Szükséges lenne még egy duplikációszűrés[8], valamint a feldolgozásban következő lépésként a mondatrabontás és tokenizálás, mely utóbbi a már említett e-magyar rendszer első lépését adja. Ezek a műveletek túlmutatnak a jelenlegi cikk célján, így nem tárgyaljuk őket. A következő fejezetben az index és a letöltött szövegek jellemzőit, valamint a letöltés tapasztalatait foglaljuk össze.

3.2. Tapasztalatok

A letöltés során fel kell készülni hálózati hibákra, valamint az adat különböző hiányosságaira is. Egyszeri tesztekből könnyen kideríthető, hogy nagyon lényeges szempont az ütemezés kérdése, mivel egy szálon maximum 50kb/s-re van limitálva a kapcsolat sebessége, valamint a kapcsolatok élettartalma is korlátozott. Ilyenkor a program úgy érzékeli, hogy a másik fél bontotta a kapcsolatot és a túlterhelés elkerülése végett automatikusan egy kis időt várakozik az újracsatlakozás előtt.

A többszálú futtatáshoz a legegyszerűbb módszer a *GNU parallel* [8] használata. Így akár több gépen, párhuzamosan több szálon is tudjuk folytatni a letöltést. A feladat jellege miatt nem szabad ragaszkodni a magonként egy szál futtatásához, mivel így nem használjuk ki az erőforrásainkat. Az optimális szám tapasztalataink szerint kb. a processzormagok 3,25-3,5 szöröse.

Mivel az indexben minden információ benne van, amely a *crawl*-olás folyamatával kapcsolatos, ezért ajánlott a letöltendő oldalakat *a priori* megszűrni a rendelkezésre álló mezők alapján:

[8] A duplikációszűrés műveletének elvégzésére ígéretes program az Onion [7], mely a `http://corpus.tools/` oldalról elérhető.

– robots.txt: a crawler működéséből adódóan minden domainen először a szabványos robots.txt-t keresi, mely nem tartalmaz számunkra használható információt.
– HTTP 3xx, 4xx, 5xx kódok: nagyon sok weboldal alkalmaz átirányítást, melyek külön elemként jelennek meg az indexben, de szöveges tartalommal nem rendelkeznek.
– mime-típus: a leszedett tartalmak sokfélék lehetnek, számunkra jelenleg csak a HTML formátumú adat értékes, így érdemes erre is szűrni.

Az így megszűrt indexbeli találatok sem jelentenek garanciát arra, hogy az adott kulcson elérhető tartalom, és valóban használható lesz számunkra. Ezért további, *a posteriori* szűrőket vezettünk be a letöltött tartalom vizsgálatára, melyek már nem az indexben található információkon alapulnak. Így külön választottuk az alábbi hibatípusokat, melyek alapján a letöltött oldalak eldobásra kerültek:

– hossz túl rövid: ha a tartalom nem volt legalább 13 karakter hosszú („<html></html>”)
– dekódolási hiba: nem helyes UTF-8 tartalom
– jól formáltsági hiba: nem jól formált XML tartalom
– nem található kulcs: az adott oldalra mutató kulcson nem található tartalom
– nincs szöveg: a sablonszűrés után nem maradt szöveges tartalom
– egyéb hibák: ide tartoznak a hálózati hibák és egyéb nem várt jelenségek (melyek többszöri próbálkozás után is fennállnak)

Bár a fent felsorolt hibák egy része javítható – például a helyes karakterkódolás meghatározásával –, jelenleg mégis az eldobásuk mellett döntöttünk, hogy felmérjük az ilyen hibák arányát, valamint el akartuk kerülni, hogy külön kelljen választani azokat az oldalakat, ahol a hibás kódolás miatt egyes szövegrészek egymástól eltérő karakterkódolásban vannak. Azokban az esetekben, ahol az indexben található elem nem volt megtalálható az API által a felhőben, külön választottuk azt, későbbi vizsgálat céljából.

4. Eredmények

A fenti szűrések elvégzése után is sok olyan eset fordult elő, ahol az *a priori* szűrés nem volt elég hatékony. Például a hiányzó mezők, valamint a szerveren rosszul feltüntetett adatok miatt sok elem az *a posteriori* szűrésen került kiszűrésre. A 2. táblázatban az *a posteriori* szűrés eredménye látható.

Mivel az Amazon 50kb/s-ben korlátozza az egy kapcsolatra jutó sávszélességet, ezért több szálon kellett futtatni a letöltést. Az összesen 64 processzormagon futtatott letöltés esetében a teljes letöltés kb. 5 napig tartott. A letöltési időt növelte, hogy az egyes HTTP kérések nem voltak újrahasznosíthatóak, mivel az Amazon bizonyos idő után mindenképpen felbontotta őket, amit a program úgy érzékelt, mintha időtúllépés miatt megszakadt volna a kapcsolat, és ezért

típus	darab	%
letöltött oldal	73 187 167	54.723988
nincs szöveges tartalom	60 294 819	45.084037
hossz túl rövid	189 006	0.141325
kulcs nem található	44 240	0.033079
dekódolási hiba	18 446	0.013793
jól formáltsági hiba	4 718	0.003528
egyéb hiba	208	0.000156
kitömörítési hiba	126	0.000094
összesen	133 738 730	100

2. táblázat. A különféle elemek típusai az *a posteriori* szűrés után

kis várakozás után mindig újra létrehozta a kapcsolatot. Így nem a processzorok teljesítménye vagy a háttértár sebessége jelentette a nehézséget.

Az egyes domaineken belüli oldalak eloszlása hatványeloszlást mutatott. A 10 leggyakoribb domain a 3. táblázatban látható. Ezek nagy része webáruház, melyben többnyire rövid, párszavas termékleírások találhatók egymás alatt. Ezzel magyarázható többek között a szöveges tartalmak alacsony aránya a letöltött oldalakhoz képest.

domain	darab	%
arukereso.hu	2 274 806	1,53
kirakat.hu	2 136 393	1,44
munkatarsaim.hu	2 094 399	1,41
docplayer.hu	2 058 703	1,39
index.hu	2 006 132	1,35
conrad.hu	1 947 660	1,32
lira.hu	1 772 434	1,20
zoover.hu	1 647 372	1,11
olcsobbat.hu	1 607 617	1,08
globalplaza.hu	1 602 395	1,08

3. táblázat. A tíz leggyakoribb domain gyakorisága és százalékos aránya

Összességében az elkészült korpusz 73 187 167 dokumentumból származó, 52 114 233 (nem tokenizált) szövegszóból és 1 396 844 paragrafusból áll, amely a 2.1. fejezetben bemutatott korpuszok tartalmánál jóval kisebb. Ez a szám a duplikációmentesítés után várhatóan még csökkenni fog. A létrejött korpusz előnye a többi felsorolt korpuszhoz képest, hogy reprodukálható, valamint rövid idő alatt előállítható mégis kellően nagy mennyiségű szövegből áll. A jövőben jó alanya lehet a sablonszűrő algoritmusok tesztelésének, valamint folyamatos bővítésével mérete is egyszerűen tovább növelhető.

5. Összegzés

A cikkben bemutattuk a kurrens webarchiválásra szolgáló technológiákat, melyekkel könnyen létrehozhatóak saját, szabványos eszközökkel feldolgozható webarchívumok és azokból kinyerhető korpuszok is. Megvizsgáltuk, hogy a szabadon elérhető, az Amazonon tárolt Common Crawl archívum .hu domainből származó része mennyire alkalmas korpusz építésére. Az előállításhoz szükséges eszközöket szabadon elérhetővé tettük[9], mellyel egy új, szabadon elérhető és reprodukálható magyar nyelvű korpuszt hoztunk létre. Habár a korpusz mérete nem haladja meg az elérhető korpuszokét, fontos tulajdonsága, hogy bármikor reprodukálható. A reprodukálhatóság előnye pedig, hogy a szöveget feldolgozó eszközök fejlődésével egy szinttel korábbról kezdődően elvégezhető a korpusz újbóli létrehozása – a fejlett eszközöknek köszönhetően – jobb minőségben.

A jövőbeli terveink között szerepel a hibák javítása, illetve a későbbi crawlok hozzáadás a korpuszhoz. További terv, hogy megvizsgáljuk, hogy mennyi duplikáció van a kinyert szövegben, mert az ilyen duplikációk nagyban torzítják a statisztikai méréseket. Végső célunk, hogy a CommonCrawl archívumtól a feldolgozott korpuszig vezető teljes út lefedése – mindenki számára elérhetően, ahol lehet, meglevő eszközökkel –, mely a későbbiekben az felhasznált eszközök fejlődésével újrajátszható marad.

Hivatkozások

1. Váradi, T., Simon, E., Sass, B., Gerőcs, M., Mittelholcz, I., Novák, A., Indig, B., Prószéky, G., Farkas, R., Vincze, V.: Az `e-magyar` digitális nyelvfeldolgozó rendszer. In Vincze, V., ed.: XIII. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY 2017), Szeged, Szegedi Tudományegyetem Informatikai Intézet, Szegedi Tudományegyetem Informatikai Tanszékcsoport (2017) 49–60
2. Halácsy, P., Kornai, A., László, N., András, R., Szakadát, I., Viktor, T.: Creating open language resources for hungarian. In N, C., ed.: Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004). (2004) 203–210
3. Váradi, T.: The Hungarian National Corpus. In: Proceedings of the Third International Conference on Language Resources and Evaluation (LREC-2002) European Language Resources Association, Paris, ELRA (2002) 385–389
4. Oravecz, C., Váradi, T., Sass, B.: The Hungarian Gigaword Corpus. In Calzolari, N., et al., eds.: Proceedings of the 9th International Conference on Language Resources and Evaluation, May 26-31, 2014, Reykjavik, Iceland, ELRA (2014) 1719–1723
5. Endrédy, I.: Nyelvtechnológiai algoritmusok korpuszok automatikus építéséhez és pontosabb feldolgozásukhoz. PhD thesis, PPKE-ITK, Budapest (2016) PhD disszertáció.
6. Endrédy, I., Novák, A.: More Effective Boilerplate Removal - the GoldMiner Algorithm. Polibits - Research journal on Computer science and computer engineering with applications (48) (2013) 79–83
7. Pomikálek, J.: Removing boilerplate and duplicate content from web corpora. PhD thesis, Masaryk university, Faculty of informatics, Brno, Czech Republic (2011)

[9] `http://github.com/ppke-nlpg/commoncrawl-downloader`

8. Tange, O.: Gnu parallel - the command-line power tool. ;login: The USENIX Magazine **36**(1) (2011) 42–47