Főnévi csoportok azonosítása magyar-angol párhuzamos korpuszban

Recski Gábor, Varga Dániel, Zséder Attila, Kornai András

BME Média Oktató és Kutató Központ, e-mail:
{recski,daniel,zseder,kornai}@mokk.bme.hu

1. Bevezetés

Cikkünkben egy magyar-angol szövegfeldolgozó rendszert mutatunk be. Elsőként a maximális főnévi csoportok magyar, illetve angol nyelvre történő azonosítását végző `hunchunk` komponenst írjuk le. A 3. részben egy szótárépítő módszert ismertetünk, a 4. részben pedig leírjuk korpuszfeldolgozó rendszerünk néhány technikai részletét, melyek lehetővé teszik, hogy nagy mennyiségű nyers kétnyelvű szöveg birtokában hatékonyan – akár több szerveren párhuzamosan – végezzük el elemzett bikorpusz építését, és az adatok webes mondattárunkba integrálását. További terveinkről az 5. részben számolunk be.

A cikkben bemutatott kétnyelvű kísérletekhez a Hunglish Korpusz [1] mondatszinten párhuzamosított, magyar-angol nyelvű bikorpuszt használtuk. A korpuszban szépirodalom, jogszabályok szövegei, hírlapok és magazinok cikkei, filmszövegek, szoftverdokumentációk, valamint pénzügyi jelentések találhatók. A cikk további részében bemutatott elemzési eljárások elvégzését követően a Hunglish Korpuszról az 1. táblázatban látható statisztikát készíthetjük.

1. táblázat. A Hunglish korpusz számai

nyelv	token	típus	tő-típus	mondat	NP
magyar	31.4M	941k	342k	2.07M	7.6M
angol	37.1M	311k	248k	2.07M	5.2M

Magyar szövegre a morfológiai egyértelműsítést és tövezést a `hundisamb` eszköz végezi. Ez a `hunmorph` morfológiai elemző által felajánlott elemzések közül választ, a `hunpos` HMM-alapú morfológiai címkéző algoritmust alkalmazva. A hunmorph-ot ehhez tőkitaláló üzemmódban használja, amely heurisztikus elemzési javaslatokkal él, ha az elemzés a szótárában megtalálható szavakra nem vezethető vissza. A hunpos címkéző működéséhez szükséges modelleket magyar nyelvre a Szeged Treebank [2], angol nyelvre a Penn Treebank [3] segítségével tanítottuk.

Angol nyelvre a hundisamb eszközt nem alkalmazhattuk, mert a hunpos angol modellje Penn Treebank címkéket bocsát ki, amelyek közvetlenül nem feleltethetők meg a hunmorph angol morfológiai címkéinek. (Terveink között szerepel ennek a kellemetlen inkonzisztenciának az orvoslása.) Itt az angol tövezőnk

által javasolt alternatívák közül mindig a legrövidebbet választottuk. Szószintű párhuzamosításhoz és fordításhoz a legrövidebb lehetséges szótő és a Penn Treebank címke együttese jól használható mint a token normálformája, bár időnként nem teljesen helyes, pl. a *grind* és *ground* főnevek normálalakja e heurisztika szerint egybeesik.

2. Főnévi csoportok azonosítása

A morfológiai információk birtokában elvégezhetjük a szónál magasabb szintű egységek azonosítását. A főnévi csoportok azonosításához (NP-chunking) az itt bemutatásra kerülő `hunchunk` eszközünket [4] használjuk, mely a szegmentálási feladatot szószintű címkézési feladattá alakítva végzi. Elsősorban a szavak elemzésére támaszkodó jegyek segítségével maximum entrópia modellt tanít, majd címkézéskor a tanítókorpuszban megfigyelt átmenetvalószínűségeket figyelembe véve mondatonként azonosítja a legvalószínűbb címkesorokat.

2.1. A tanulóadatok előállítása

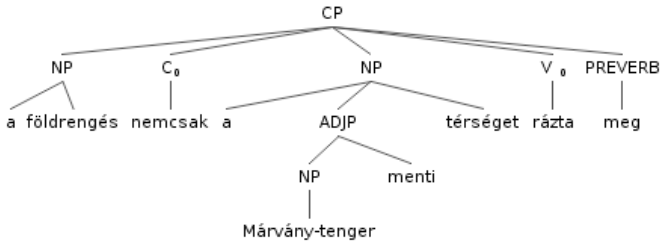
A magyar nyelvű tanulóadatokat a Szeged Treebankből nyerjük ki: a korpuszban található maximális NP-ket feleltetjük meg chunkoknak, tehát azokat a főnévi csoportokat, melyeket más NP nem dominál. Bár az NP-chunkok azonosítása a szakirodalomban leggyakrabban valamennyi minimális NP megkeresését jelenti, célszerűbbnek láttuk a fenti definíciót alkalmazni, mivel így lehetőségünk nyílik a mondatok közvetlen összetevőinek elkülönítésére és az igék argumentumszerkezetének feltérképezésére.

A tokenek címkézésekor a Start/End [5] konvenciót alkalmazzuk, mely az elterjedtebb IO és IOB konvencióknál [6] több címkét igényel, ugyanakkor lehetővé teszi többféle chunkbeli pozíció megkülönböztetését: míg az előbbi megoldások vagy egy címkével (I-NP) jelölik a chunkhoz tartozó szavakat, vagy ezen felül még a chunkot kezdő szót jelölik külön szimbólummal (B-NP), addig az általunk használt jelölés a chunkhoz nem tartozó szavakon (O) kívül négy címkét használ (B-NP, I-NP, E-NP, 1-NP), melyek rendre a chunk elején, közepén és végén álló, valamint az önmagában chunkot alkotó szavakat jelölik.

Az adatok kinyerésekor feljegyezzük azt is, hogy az adott NP-be milyen mélyen ágyazódnak további NP-k, így lehetőségünk nyílik egyfajta komplexitásfogalom alapján több chunktípust megkülönböztetni. Az effajta információk kinyerését nem tekintjük a címkéző feladatának, csupán a gépi tanulási feladatot könnyítjük meg vele: optimálisnak az a címkézés bizonyult, ahol csupán a legalacsonyabb – tehát további NP-t nem tartalmazó – chunkokat különböztettük meg (N_1) a komplexebbektől (N_2+). A fenti chunkdefiníció és címkézés eredményeképp a Szeged Treebank az 1. ábrán látható mondata a chunkkorpuszban a 2. ábra szerinti reprezentációt kapja.

A	földrengés	nemcsak	a	Márvány-tenger	menti	térséget	rázta	meg
B-N_1	E-N_1	O	B-N_2+	I-N_2+	I-N_2+	E-N_2+	O	O

1. ábra.



2. ábra.

Az angol nyelvű tanulóadatok kinyeréséhez a Penn Treebanket használjuk. Itt NP-chunknak tekintjük a maximális főnévi csoportok mellett azon prepozíciós frázisokat is, melyek tartalmaznak főnevet, nem tartalmaznak igét és nem képezik részét magasabb szintű NP-nek. Ezzel [7] definícióját követjük, melyet a szerző azzal motivál, hogy az NP és PP szerkezetek közti határt a különféle nyelvek nem ugyanott húzzák meg, illetve a két kategória számos nyelvben nem is különül el egymástól élesen. Fontosnak tartjuk megemlíteni, hogy az NP-chunk definíció mindkét nyelv esetében csupán a korpuszt előállító rendszer beállításaitól függ, így amennyiben eltérő egységeket tekintünk chunknak – így például a fent említett módon a minimális NP-ket szeretnénk azonosítani – úgy ahhoz egyszerűen állítható elő megfelelő tanítókorpusz.

2.2. A jegyek

A tanítás elsősorban szószintű jegyek alapján történik: egy szó jegyének tekintjük a szótövet és valamennyi morfológiai jegyet. A Szeged Treebank MSD-konvenció szerinti annotációját átalakítottuk a KR-formalizmusra, mivel az általunk használt morfológiai címkéző, elemző és egyértelműsítő egyaránt ezt a formátumot követi. Jegyként vettük fel az így előállított KR-kódok valamennyi elemi összetevőjét. A Penn Treebank esetében ezt nem tehettük meg, mivel az abban használatos morfológiai címkék nem kompozicionálisak, így ott a teljes címke mellett csupán annak első karakterét – mely a szófajt azonosítja – vesszük fel önálló jegyként. A szószintű jegyeket minden tokenre annak 5 szavas környezetében értékeljük ki.

Bevezettünk továbbá egy jegyet, mely egy szó adott hosszúságú környezetében az egymást követő szavak szófaji címkéinek sorozatait írja le a követ-

kező módon: ha a jegy sugarát r-rel, egy mondat i-edik pozíciójában álló szót w_i-vel, szófaji címkéjét pedig p_i-vel jelöljük, úgy, úgy bármely w_i szóra jegyként vesszük fel a $p_{i-r} \dots p_{i+r}$ sorozat összes összefüggő részintervallumát. A KR-mintákat kiválasztó jegy sugarát növelve a chunkolás F-pontszáma is nő, 3-nál magasabb sugár mellett azonban a jegyek magas száma nem teszi lehetővé a modell tanítását.

2.3. A statisztikus modell

A címkézési feladat modellezéséhez rejtett Markov Modellt (HMM, [8]) tanítottunk, melynek kibocsátási modelljét Maximum Entrópia modellből [9] nyertük. Az alábbiakban ismertetjük modellünket, és a mögötte álló statisztikai előfeltevéseket.

Jelölje $p(i, u)$ annak valószínűségét, hogy az i pozícióban álló szó az u címkét kapja. Feltételezzük, hogy $p(i, u)$ értéke kizárólag annak $w_{i-k} \dots w_{i+k}$ környezetétől függ. Ekkor $p(i, u)$ értékét $\hat{p}(i, u)$ kiszámításával becsüljük, melyet a korábban ismertetett jegyeken tanított ME modell szolgáltat. Jelölje $t(i, u, v)$ annak feltételes valószínűségét, hogy az i pozícióban álló szó u címkét kap, feltéve hogy az $i - 1$ pozícióban álló szó a v címkét kapta. Feltételezzük, hogy ez a valószínűség független i-től és a tanítókorpuszban megfigyelt feltételes relatív gyakorisággal ($\hat{t}(u, v)$) adunk rá becslést.

A címkézés során a rendszer egy adott mondatra adható legvalószínűbb címkesorozatot keresi. Ha $\hat{p}(i, u)$ csupán w_i-től függne (tehát nem számítana a környezet), akkor egy sorozat valószínűsége a feltételes függetlenségnek köszönhetően szorzatként állna elő és az alábbi képlettel lenne arányos:

$$\prod_i \frac{\hat{p}(i, u_i)\hat{t}(i, u_i, u_{i-1})}{P(u_i)}.$$

Ezen képlet maximuma, tehát a legjobb címkesorozat megtalálható a Viterbi algoritmus segítségével. Ez a modell valójában a ‘megfigyelések az állapotokban és nem az átmenetekben’ változata a Maximum Entrópia Markov Modellnek, ahogy [10] javasolja. Modellünket úgy írhatjuk le, mint ennek a modellnek az egyszerű általánosítását: megengedjük, hogy $\hat{p}(i, u)$ egy $w_{i-k} \dots w_{i+k}$ ($k > 0$) környezettől függjön és a fenti képlet segítségével becsüljük a tényleges valószínűséget. A tekintetbe vett környezet k sugara optimalizálandó paraméter, a cikkünkben említett összes feladaton az 5 sugár bizonyult optimális választásnak.

Rendszerünknek még egy szabad paramétere a nyelvi modell súlyozása. Ez standard megoldás a HMM szakirodalomban. Esetünkben a fenti képletet ez úgy általánosítja, hogy egy pozitív λ kitevőt alkalmazunk a $\hat{p}(i, u_i)$-re és a $P(u_i)$-re. A λ paramétert kisméretű részkorpuszon optimalizáltuk egy-egy feladathoz.

2.4. A magyar és angol NP-chunking kiértékelése

A főnévi csoport azonosító kiértékeléséhez NP-korpuszainkat mindkét nyelven egy 1000000 token hosszúságú tanító- és egy 500000 token hosszúságú teszt-

korpuszra osztottuk véletlenszerűen. A tesztkorpuszon lefolytatott címkézések kimenetét a [11]-beli szabályokat követve értékeltük ki. Összehasonlítási alapmódszerként (baseline) magyar nyelvre minden szónak a szófaji címkéje alapján legvalószínűbb címkét osztottuk ki. A legegyszerűbb címkézési módszert követve, amely csupán az I-NP és O címkéket használja, a rendszer csupán 51.03%-os F-pontszámot ért el. Egy kevés bonyolítással - harmadikként bevezetve a B-NP címkét – az eredmény 60.37%-ra nőtt. Rendszerünk eredményei magyarra a 2. táblázatban láthatóak.

2. táblázat.

	Pontosság	Fedés	F-pontszám
baseline	60.24%	60.50%	60.37%
hunchunk	89.40%	89.97%	**89.68%**

Felhívjuk a figyelmet arra, hogy az NP chunk általunk adott, a szakirodalomban legelterjedtebbtől eltérő definíciója jelentősen hosszabb és szerkezetüket tekintve komplexebb NP-ket eredményezett, mint pl. a [11] szerinti, ún. alap NP („base NP"). Ez magyarázza a szakirodalomban szokásosan láthatónál alacsonyabb pontszámokat. Noha célunk a maximális NP-k azonosítása volt, algoritmusunkat egy minimális NP-feladaton is kipróbáltuk, hogy teljesítményét összevethessük a legkorszerűbbnek tartott statisztikus szegmentálóalgoritmusokéval. A CoNLL 2000 Shared Taskon, melynek tanuló- és tesztadata rögzített, és a szegmentálóalgoritmusok összehasonlításának standard terepeként szolgál, eszközünk 93.79%-os F-pontszámot ért el. Ez körülbelül fél százalékkal alacsonyabb, mint a modelltanításkor egy nagyságrenddel nagyobb számításigényű CRF algoritmusok eredménye: [12] 94.34%, [13] pedig 94.29% F-pontszámot publikált a feladaton. A `hunchunk` kétfajta feladaton elért eredményeit a 3. táblázat tartalmazza.

3. táblázat.

Feladat	Pontosság	Fedés	F-pontszám
max NP	79.33%	79.87%	**79.60%**
base NP	93.61%	93.85%	**93.73%**

3. Szótárépítés

Az alábbiakban egy egyszerű iteratív szótárépítő algoritmust mutatunk be, amely az együttes előfordulások aránya alapján rangsorolja a szótári tételeket. A miénkhez hasonló, úgynevezett Competitive Linkingen alapuló algoritmust

elsőként Melamed [14] publikált. Ezután bemutatjuk, hogy a Competitive Linking eljárás pontossága növelhető, ha kiaknázunk egy automatikus szószintű párhuzamosítást.

3.1. Az algoritmus

Szótárépítési eljárásunk alapja a Dice együttható néven ismert mérőszám egy magyar-angol szópár együtt-előfordulásának mértékére: ennek definícója $D = o_{he}/(o_h + o_e)$, ahol o_{he} a szópár együttes előfordulásainak száma, o_h és o_e az egynyelvi előfordulások száma. Ha egy bimondatban több előfordulás is van, akkor a két előfordulásszám mondatbeli minimumával (és nem szorzatával) járul hozzá a mondat az o_{he} mennyiséghez.

Az algoritmus első lépésként összegyűjti az összes olyan magyar-angol szópárt, amelyre D egy t küszöb felett van, ahol t az algoritmus paramétere. Ha egy szó egynél több így azonosított szótári tételben is szerepel, akkor csak a legnagyobb hasonlósági mértékűt tartjuk meg. Az iteráció kimenete az így összegyűjtött tételek halmaza. Ezután a korpusz bimondataiban összekötjük azokat a magyar-angol szópárokat, melyek megtalált szótári tételnek felel meg, és töröljük a korpuszból az összes összekötött szót. Ezen a ponton újrakezdhetjük az iterációt a Dice együtthatók kiszámításával, és joggal remélhetjük, hogy a korábbi tételek eliminálása után egyes új tételek hasonlósága a küszöb fölé lép. Az iterációt addig folytatjuk, amíg a szótár már nem bővül tovább – kísérleteinkben ehhez 10-15 iterációra volt szükség, egyre csökkenő hosszúságú iterációk mellett.

A most ismertetett eljárást ItCo-nak fogjuk nevezni az alábbiakban. Az eljárásnak megvizsgáljuk majd azt az ItCo+GIZA változatát is, amely (az alapváltozattal ellentétben) feltételezi egy szószintű párhuzamosítás meglétét. Ez a változat csak azokat az együttes előfordulásokat veszi számításba, amelyeknél a két szó között kapcsolat van a párhuzamosításban.

A szószintű párhuzamosítás építéséhez a szakirodalomban teljesen standardnak tekinthető GIZA++ és Moses eszközöket választottuk. Az IBM tanulóalgoritmusának [15] GIZA++ [16] által adott implementációja egy ún. IBM Model 5 fordítási modellt épít a tokenizált párhuzamos korpuszból, amiből egy asszimetrikus szószintű párhuzamosítást nyerhetünk ki. Ezt lépést a magyar-angol és angol-magyar fordítási irányokra egyaránt elvégezve két „félkész" párhuzamosítást kapunk. Ezeket a Moses [17] frázisalapú gépi fordítóhoz mellékelt heurisztikus algoritmus fésüli össze minél konzisztensebb szimmetrikus szószintű párhuzamosítássá.

3.2. Az eljárás kiértékelése

Méréseinkhez a Hunglish Korpusz tövezett, szószinten párhuzamosított változatát alkalmaztuk. A szavak halmazán szótárépítés előtt háromféle szűrést is végeztünk: elhagytuk a funkciószavakat, azokat a szavakat, amelyek nem szerepeltek legalább 10-szer a korpuszban, továbbá azokat a szavakat (szótöveket) is, amelyek nem szerepeltek magyar, illetve angol tövezett gyakorisági

szótárainkban. (Előbbinek a forrása a Szószablya Webkorpusz, utóbbié a Google 1T webkorpusz [18], mindkettőt a `hunmorph` eszközzel [19] töveztük.) A szótárépítés elvégzése után pedig elhagytuk a nagy kezdőbetűs tételeket is, hiszen ezek nagy pontossággal megfeleltek a tulajdonneveknek.

Az automatikusan létrehozott szótárakról automatikus eszközökkel igazán pontos minőségi információkat kinyerni nem lehet. De hogy már az előzetes kísérletek során fogalmat nyerhessünk a szótárak relatív minőségéről különböző paraméterbeállítások mellett, először mégis a Vonyó Attila szótárával való százalékos átfedésüket vizsgáltuk. Ezen mérések alapján úgy választottuk meg a szótárépítő algoritmus Dice paraméterét (0.095-nek), hogy egyensúlyt valósítsunk meg a szótár pontossága és mérete között. Az optimálisnak talált paraméterbeállítás mellett 21846 méretű szótárunk tételeinek 53.9%-a szerepelt a Vonyó-szótárban, amely arány 71.5%-ra nőtt, ha a szavaknak csak az 5 hosszú kezdőszeleteit illesztettük. Hangsúlyozzuk, hogy ez globális pontossági mértékként félrevezető, amennyiben a Vonyó-szótárban nem szereplő tételek többsége is legitim találat.

A paraméterhangolás után elvégeztük a manuális kiértékeléseinket. A hibafajták között nem meglepő módon a domináns az volt, amikor a szópár képzőtől eltekintve helyes volt. Ez előállhat akkor, amikor a magyar és angol szöveg különböző szófajú konstrukcióval fejez ki egy adott fogalmat, illetve ha a két tövező másként dönt egy képzett szó lexikalizált mivoltáról, pl. *vallásos-religion, szerencse-lucky, forgatás-rotate, szökött-escape, továbbfejleszt-development.* A manuális kiértékeléskor ezt a hibaosztályt külön jelöltük. Kétféle pontosságmértéket alkalmaztunk egy szótár minőségének manuális számszerűsítésére: a teljesen helyes tételek arányát, illetve a képzési hibától esetlegesen eltekintve helyes tételek arányát, amit nemhelytelen-nek neveztünk.

3.3. Eredmények

A kiértékelés alapjának [20]-gyel azonos módon a GIZA++ IBM Model 5 szótárépítőjét választottuk, amely a Model 5 fordítási modellből nyeri ki a szótárat. A rendszer minden szótári tételhez egy 0 és 1 közötti konfidenciaértéket ad. Ezek csökkenő sorrendje szerint rendezzük a szótári tételeket, így tetszőleges arányt megcélozhatunk méret és pontosság között. Egy általunk épített szótárral való pontossági összehasonlításkor a GIZA++ szótár akkora méretű kezdőszeletét választottuk, amekkora az összehasonlítandó szótár. A baseline szótár építésekor ugyanazokat a szűrőket alkalmaztuk, mint saját szótáraink esetében. Az eredmények szótáranként 200 véletlen minta vétele alapján kiértékelve a 4. táblázatban láthatók.

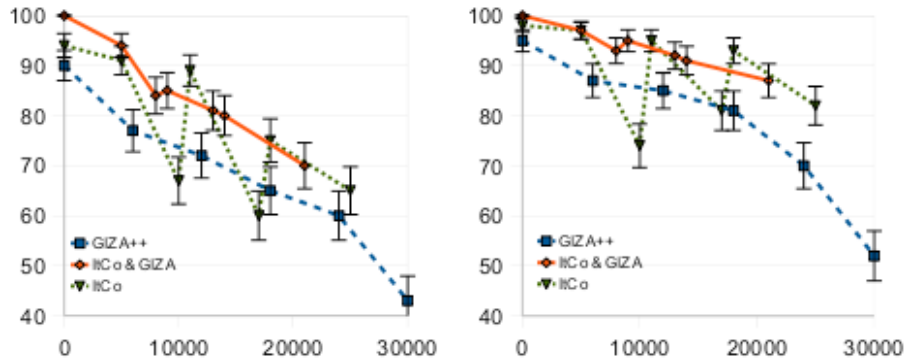
Az épülő szótár mérete az algoritmus paramétereitől függ. Az algoritmus futása jól elkülönülő iterációkból áll (10-15 egyre csökkenő méretű iteráció). Egy iteráción belül fokozatosan csökken a konfidencia és a tényleges pontosság is. Két iteráció között azonban felugrik a konfidencia és pontosság, hiszen a helyesen azonosított szótári tételeknek megfelelő szópárok elhagyása csökkenti a félrevezető kollokációk halmazát. Ez egy fűrésszerű pontossági grafikonhoz ve-

4. táblázat.

	Baseline	ItCo	Baseline	ItCo+GIZA
helyes %	68.5	77.0	69.2	81.5
nemhelytelen %	76.5	87.5	76.7	92.5
szótáméret	25422	25422	21846	21846

zet, ha az x-tengelyen azt ábrázoljuk, hogy a tétel hanyadikként lett azonosítva, az y-tengelyen pedig a simított pontosságot.

Ahhoz, hogy bemutathassuk ezeket a pontossági grafikonokat, mintavételezésre volt szükség, hiszen 25000 szótári tétel manuális ellenőrzése túlságosan időigényes feladat. A grafikon egy adatpontját ezért a következő módon hoztuk létre. Az x pozíció mintavételezéséhez véletlenszerűen kiválasztottunk a szótár (x,x+1000) intervallumából 100 tételt. Ezeket manuálisan klasszifikáltuk a már említett (helyes, képzéstől eltekintve helyes, helytelen) kategóriákba. Ez az adott x szótárpozícióhoz két különböző százalékos pontossági értéket rendelt: az egyik a helyes tételek aránya, a másik a nemhelytelen tételeké.



3. ábra.

A 3. ábrán látható, hogy a mintavételezés a GIZA++ által épített szótárak esetében szabályos lépésenként történt, a saját szótáraink esetében viszont nem. Ennek oka a grafikonok már említett fűrész-alakja. A lépésközt úgy választottuk, hogy az első két, 1000-nél még nagyobb méretű fűrészfog (azaz iteráció) belsejében két mintavételezési pont legyen: az iteráció elejéről illetve végéről. Az első, domináns méretű iterációnak a közepéről is mintavételezünk. Az ábrákról leolvasható, hogy lineárisan interpolálva a mintavételezési pontokat, a GIZA+ItCo módszer pontossága a GIZA módszeré felett van minden pontban, a ItCo módszeré pedig a legtöbb pontban.

4. Implementáció

Ebben a szakaszban szövegfeldolgozó rendszerünk néhány műszaki részletéről számolunk be.

4.1. Keretrendszer

Elsősorban az a keretrendszer érdemel említést, amelyet az adatok feldolgozására kiépítettünk. Ennek feladata az egyes feldolgozó modulok (pl. tokenizálás, szófaji elemzés) hatékony futtatása nagy méretű adathalmazokon. A rendszer nagyon rugalmas keretet ad az általa futtatott moduloknak, nem kötelezi el magát például abban sem, hogy milyen programozási nyelven kell implementálnunk azokat.

A keretrendszer használatához az elvégzendő feladatok irányított aciklikus gráfját kell definiálnunk, megadva, hogy a csúcsokhoz tartozó feladatok milyen parancsnak felelnek meg. A keretrendszer feldolgozandó fájlok egy halmazára alkalmazza ezt a pipeline-t vagy valamely kijelölt részgráfját, egy standardizált struktúrájú könyvtárhierarchiát hozva létre. Két specializált szolgáltatást nyújt a rendszer, amelyek gyorsítják a feladat elvégzését, ezek akár egyszerre is kiaknázhatóak:

- Párhuzamosítás: A rendszer képes felhasználni a feladatok párhuzamos elvégzéséhez egy számítógép-klasztert, a klaszterben részt vevő számítógépek egyes processzorait párhuzamosan terhelve. Ehhez csupán arra van szükség, hogy a klaszter egyes tagjai hozzáférjenek az adatokat és modulokat tartalmazó fájlrendszerhez. Az ütemezés alapegysége a dokumentum, tehát egyetlen nagyméretű dokumentumot már nem tördel kisebbekre az ütemező.
- Daemon: A hunchunkhoz és hunnerhez hasonló gépi tanuló rendszerek statisztikus modelleket tartalmazó, sok megabájtos erőforrásfájlokat olvasnak be induláskor. Ezért ha sok kis dokumentumra futtatnánk ezeket, akkor a futásidő nagy részét inicializálással töltenék. A keretrendszer daemon üzemmódja ezt a problémát úgy orvosolja, hogy a munka kezdetén egyetlen alkalommal indítja csak el a címkéző/szegmentáló programot, majd az ütemezett fájlokat unix socketokon keresztül kommunikálva egymás után küldi el annak. A „becsomagoláskor” a keretrendszer a daemonként elindított programról nagyon kevés előfeltevéssel él. Ez a megoldás alkalmazandó akkor is, ha a címkéző/szegmentálókat például webszolgáltatás részeként kívánjuk alkalmazni.

A Hunglish Korpusz építését újraimplementáltuk a keretrendszerben, tehát az elemzési lépések kiindulópontja lehet nyers, formázatlan szöveg két nyelven. A megfelelő elemzési lépések elvégzése után a Hunglish Mondattár webes keresőrendszer indexelőjéhez vagy a Moses fordítórendszer modellépítőjéhez vagy dekóderéhez továbbíthatóak a feldolgozott adatok.

4.2. Huntag

A hunchunk a korábban publikált hunner rendszerhez [21] algoritmikusan nagyon hasonló – egyetlen különbségük, hogy a hunchunk a szegmentumok közti átmenet-valószínűségeket tanulja. A hunner rendszert ezért újraimplementáltuk, és a két szegmentálót egyetlen közös, huntag-nek nevezett eszközben valósítottuk meg, amelyet csak a jegy-számításért és paraméterezésért felelős leírófájlok adaptálnak egyik vagy másik feladathoz. A reimplementáció nem volt komoly hatással a hunner pontosságára, 96.35%/95.05%-ről 96.53%/94.81%-re változott a Szeged NER fejlesztő, illetve tesztelő adathalmazain.

5. További terveink

Elsődleges további tervünk olyan eljárás publikálása, és teljesítményének számszerűsítése, amely az azonosított maximális NP-ket párhuzamosítja a kétnyelvű szöveg bimondataiban. (Ilyen jellegű rendszert először Pohl [22] publikált magyar nyelvre, magyar-angol fordítómemória építése céljából.) Bár számszerűsíthető adataink a kézirat leadásakor még nincsenek, azt gondoljuk, hogy a maximális NP-k közt jóval nagyobb arányú az 1-1 párhuzamosság mint a szavak vagy alap NP-k közt, és hogy az NP-párhuzamosítási feladat hatékony megoldása nemcsak a gépi fordítást, hanem a mondatok argumentumszerkezetének megértését is segíteni fogja.

A keretrendszer és a huntag rendszer más technológiáinkhoz hasonlóan szabad forráskódúak. A cikk írásának időpontjában a :pserver:anonymous:anonymous@cvs.mokk.bme.hu:/local/cvs cvs-szerver tcg, illetve huntaggers moduljaiként már bárki számára elérhetőek, de célunk, hogy a rendszert a konferencia idejére megfelelő minőségű dokumentációval is ellássuk.

Hivatkozások

1. Varga, D., Németh, L., Halácsy, P., Kornai, A., Trón, V., Nagy, V.: Parallel corpora for medium density languages. In: Proceedings of the Recent Advances in Natural Language Processing 2005 Conference, Borovets. Bulgaria (2005) 590–596
2. Csendes, D., Csirik, J., Gyimóthy, T., Kocsor, A.: The Szeged Treebank. In: Lecture Notes in Computer Science: Text, Speech and Dialogue. (2005) 123–131
3. Marcus, M.P., Santorini, B., Marcinkiewicz, M.A.: Building a large annotated corpus of english: The Penn Treebank. Computational Linguistics **19** (1994) 313–330
4. Recski, G., Varga, D.: Magyar főnévi csoportok azonosítása. Általános Nyelvészeti Tanulmányok (2010)
5. Uchimoto, K., Ma, Q., Murata, M., Ozaku, H., Isahara, H.: Named entity extraction based on a maximum entropy model and transformation rules. In: ACL '00: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics, Morristown, NJ, USA, Association for Computational Linguistics (2000) 326–335
6. Sang, E.F.T.K., Veenstra, J.: Representing text chunks. In: EACL. (1999) 173–179

7. Koehn, P., Knight, K.: Feature-rich statistical translation of noun phrases. In: In Proc. of the 41st Annual Meeting of the ACL. (2003) 311–318
8. Rabiner, R.L.: A tutorial on Hidden Markov Models and selected applications in speech recognition. In: Proc. IEEE. Volume 77. (1989) 257–286
9. Ratnaparkhi, A.: Maximum entropy models for natural language ambiguity resolution. Technical report (1998)
10. Mccallum, A., Freitag, D., Pereira, F.: Maximum entropy markov models for information extraction and segmentation. In: Proc. 17th International Conf. on Machine Learning. (2000) 591–598
11. Sang, E.F.T.K., Buchholz, S., Sang, K.: Introduction to the CoNLL-2000 shared task: Chunking (2000)
12. Sun, X., Morency, L.P., Okanohara, D., Tsujii, J.: Modeling latent-dynamic in shallow parsing: a latent conditional model with improved inference. In: COLING '08: Proceedings of the 22nd International Conference on Computational Linguistics, Morristown, NJ, USA, Association for Computational Linguistics (2008) 841–848
13. Sha, F., Pereira, F.: Shallow parsing with conditional random fields. In: NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, Morristown, NJ, USA, Association for Computational Linguistics (2003) 134–141
14. Melamed, I.: (Empirical methods for exploiting parallel texts)
15. Brown, P.F., Pietra, V.J.D., Pietra, S.A.D., Mercer, R.L.: The mathematics of statistical machine translation: parameter estimation. Computational Linguistics **19** (1993) 263–311
16. Och, F.J., Ney, H.: A systematic comparison of various statistical alignment models. Computational Linguistics **29** (2003) 19–51
17. Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., Herbst, E.: Moses: Open source toolkit for statistical machine translation. In: Proceedings of the ACL'07, The Association for Computer Linguistics (2007)
18. Brants, T., Franz, A.: Web 1t 5-gram corpus version 1. Technical report, Google Research (2006)
19. Trón, V., Gyepesi, G., Halácsy, P., Kornai, A., Németh, L., Varga, D.: Hunmorph: open source word analysis. In: Proceedings of the ACL 2005 Workshop on Software. (2005)
20. Karlgren, J., Sahlgren, M.: Automatic bilingual lexicon acquisition using random indexing of parallel corpora. Natural Language Engineering **11** (2005) 327–341
21. Varga, D., Simon, E.: Hungarian named entity recognition with a maximum entropy approach. Acta Cybernetica **16** (2006) 293–301
22. Pohl, G.: English-hungarian np alignment in metamorpho tm. In: Proceedings of the EAMT. (2006)