Magyar főnévi csoportok azonosítása

Recski Gábor[1], Varga Dániel[2]

[1] MTA SZTAKI Nyelvtechnológiai Kutatócsoport, recski@sztaki.hu
[2] BME Média Oktató és Kutató Központ, daniel@mokk.bme.hu

Jelen cikkünk egy NP-felismerő (NP-chunker) alkalmazást mutat be, mely magyar nyelvű főnévi csoportok azonosítását teszi lehetővé felügyelt gépi tanulás segítségével. Az **1.** fejezetben az *NP-chunk* és az *NP-chunkolás* fogalmát mutatjuk be, áttekintve a szakirodalomban használatos definíciókat, az elterjedtebb technológiákat és az eljárás alkalmazási területeit. A **2.** fejezetben a felügyelt tanuláshoz szükséges tanítóadatokat, azaz a magyar NP-korpusz létrehozását ismertetjük. A **3.** fejezet bemutatja a `hunchunk` gépi tanuló rendszer működését a modell tanításától az NP-felismerésig. A **4.** fejezet a rendszer kiértékelésével foglalkozik. Az utolsó, **5.** fejezetben röviden beszámolunk folyamatban lévő munkánkról, melynek célja egy magyar-angol párhuzamos NP-korpusz létrehozása.

1. Chunk, chunkolás, NP-felismerés

1.1. Meghatározás

Sem a *chunk*, sem a *chunkolás*) kifejezés nem rendelkezik általánosan elfogadott definícióval a számítógépes nyelvészeti szakirodalomban. A chunk kifejezést elsőként Abney (1991:1) használja, aki a mondat olyan, egymással át nem fedő egységeit nevezi így, melyek „egyetlen tartalmas szóból és az őt körülvevő funkciószavakból áll". Elsősorban Gee–Grosjean (1983) munkájára alapozva, akik a mondat pszicholingvisztikai egységeinek megnevezésére a *performanciaszerkezet* (*performance structure*) fogalmát vezetik be, Abney a chunkokat olyan egységekként kezeli, melyek nem szükségszerűen esnek egybe mondattani összetevőkkel. A chunkolás témakörében végzett újabb kutatások ezzel szemben egyöntetűen olyan meghatározásokkal élnek a chunk fogalmára, melyek lehetővé teszik, hogy egy mondat chunkjait annak teljes elemzési fájából egyértelműen előállíthassuk. A gyakorlatban ez azt jelenti, hogy valamennyi chunk-típust szintaktikai frázisok valamely csoportjának feleltetünk meg és ezzel a chunkolás a teljes mondattani elemzés részfeladatává válik. Egy minden szintaktikai kategóriára kiterjedő definícióhalmazt találunk a CoNLL 2000 (Tjong Kim Sang–Buchholz 2000) versenyfeladat[1] leírásában, ennek alapján nyerik ki a chunkolási mintát a Penn Treebank (Marcus et al. 1993) szintaktikailag annotált korpuszból.

[1] A *versenyfeladat* (shared task) azt jelenti, hogy egy konkrét feladatot, jelen esetben az angol NP, VP, PP, stb. chunkok megtalálását, egységes adathalmazon és sztenderdizált kiértékelési eljárás mellett a tudományos közösség elé tárnak, így az annak megoldására javasolt különböző eljárások pontosan összemérhetővé válnak.

A chunkolás legelterjedtebb formája az NP-chunkok azonosítása. Az NP-chunkolás témakörében született egyik legismertebb mű szerzői Ramshaw és Marcus (1995), akik az *alap NP-k* (más néven *baseNP-k*, *nem-rekurzív NP*-k vagy *minimális NP*-k) azonosításával foglalkoztak, értve ez alatt azokat a főnévi csoportokat, melyek nem tartalmaznak másik főnévi csoportot. A CoNLL 2000 feladatához is ezt a meghatározást használta fel Tjong Kim Sang és Buchholz, chunkolási feladatukat az eredeti, Ramshaw és Marcus által megadott módon tűzték ki, és ez a feladat szolgál mindmáig terepként a különböző gépi tanulási algoritmusok összehasonlítására.

Magyar nyelvű NP-chunkok azonosításával foglalkozik Váradi (2003) és Prószéky et al. (2004). Az általunk bemutatandó rendszer ezektől egyrészt abban különbözik, hogy nem szabályalapú, hanem statisztikai módszerekkel azonosítja a chunkokat (e módszereket a 3. fejezetben részletesen bemutatjuk), másrészt pedig az NP-chunknak a szokásostól eltérő definícióját használja (ezt a következő fejezetben részletezzük). A magyar NP-chunkok azonosítására alkalmas eszközöket – beleértve az itt bemutatandó `hunchunk` eszköt is – Miháltz sztenderdizált körülmények között hasonlítja össze (2011).

1.2. Alkalmazások

A teljes mondattani elemzés (*parsing*) sokkal alacsonyabb pontossággal végezhető, mint a chunkolás, így utóbbi lehetővé teszi, hogy kevesebb, de megbízhatóbb információt nyerjünk ki egy mondat szerkezetéről. Az NP-chunkok azonosítása egyes információ-kinyerési feladatokhoz is hozzájárul, amennyiben a főnévi csoportokkal egyszersmind a mondatban szereplő entitásokat is azonosítja.

2. A tanulóadatok előállítása

A legelterjedtebb NP-chunkoló eszközökhöz hasonlóan az általunk készített rendszer is *felügyelt tanulás*ra (supervised learning) épül, azaz az alkalmazás egy manuálisan előállított tökéletes minta alapján, statisztikai módszerekkel tárja fel a szavak egyes tulajdonságai és chunkhoz tartozásuk közti összefüggéseket. (A rendszer működésének részleteit a 3. fejezetben mutatjuk be.) Ezért első lépésként szükségünk van egy ilyen ún. tanulóadathalmaz létrehozására egy mondattanilag annotált korpusz segítségével.

2.1. A feladat

Bár a szakirodalomban NP-chunkolás alatt általában az alap NP-k megtalálását értik, mi egy ezzel gyakorlatilag ellentétes definíciót választunk, amennyiben NP-chunknak tekintünk minden olyan szósorozatot, mely a mondat elemzési fájában NP-t alkot és ezt az NP-t nem tartalmazza magasabb szintű főnévi csoport (ezeket fogjuk *maximális NP*-knek nevezni). Ez a definíció lehetővé teszi, hogy a chunkolással a mondat közvetlen összetevőit különítsük el és a mondatban szereplő igék vonzatkeretét feltérképezzük, mely a gépi fordításban

különös jelentőséggel bír. Ezen túl a maximális NP-k azonosítása az információkinyerésben is hasznos lehet, amennyiben a mondatokban szereplő főneveket azok összes bővítményével együtt nyerjük ki. Fontosnak tartjuk megemlíteni, hogy a használt NP-chunk definíció csupán a korpuszt előállító rendszer beállításaitól függ, így amennyiben eltérő egységeket tekintünk chunknak – például a fent említett módon az alap NP-ket szeretnénk azonosítani – úgy ahhoz egyszerűen állítható elő megfelelő tanítókorpusz. A jelen cikkben bemutatott rendszer tehát bármilyen módon definiált NP-chunk azonosítására alkalmas, választásunk jelentősége abban rejlik, hogy a rendszer – későbbiekben részletesen ismertetett – paramétereit, így különösen a tanításhoz használt jegyek összetételét úgy választottuk meg, hogy a maximális NP-k azonosításban a lehető legjobb eredményt érje el.

Tanulóadataink forrása az 1.43 millió szóból álló Szeged Treebank korpusz (Csendes et al. 2005), mely különböző műfajú (szépirodalom, újságcikkek, jogszabályi szövegek, szoftverdokumentációk, stb.), morfológiailag annotált és mondattanilag elemzett szövegekből áll. Egy Treebank a benne szereplő mondatok teljes elemzési fáját tartalmazza, így az NP-k azonosításához szükségesnél bővebb szerkezeti információkat is, melyekre a NP-korpusz előállításakor nincsen szükségünk. Az általunk elvégzett eljárás lényege, hogy a Treebank-ben található elemzési fákat bejárjuk, az azokban található szavakat pedig a rendelkezésre álló morfológiai információkkal együtt a korpuszhoz adjuk, feljegyezve azt is, hogy részét képezik-e maximális NP-nek, azaz a mi definíciónk szerinti NP-chunknak.

Az NP-felismerési feladatot címkézési feladatként oldjuk meg, ami azt jelenti, hogy egy chunkolás alapján a mondat minden szavához címkét rendelhetünk, mely azt írja le, hogy az adott szó részét képezi-e NP-nek. A címkézés legelterjedtebb – és a CoNLL 2000 feladatban is használatos – módja az ún. IOB konvenció, mely három címkét különböztet meg: az NP-k első szava a `B-NP`, többi szava az `I-NP` címkét, az NP-hez nem tartozó szavak pedig az `O` címkét kapják (Tjong Kim Sang–Veenstra 1999).

Ez a jelölés nem különbözteti meg az NP végén álló szavakat az NP közepén állóktól, az önmagukban főnévi csoportot alkotó szavakat pedig ugyanúgy jelöli, mint az NP-t kezdő szavakat. Feladatunkban célszerűbbnek bizonyult az 5 címkét használó Start-End konvenciót használni, mely a fentieket pontosítva az NP-végző tokeneket `E-NP` címkével, az egyszavas NP-ket pedig `1-NP` címkével jelöli (Uchimoto et al. 2000). Az effajta címkézést megvalósító rendszerünknek biztosítania kell, hogy szabálytalan – azaz NP-chunkolásnak meg nem feleltethető – címkesorok ne jöhessenek létre; például az `I-NP` címkét nem követheti legálisan a `B-NP` címke.

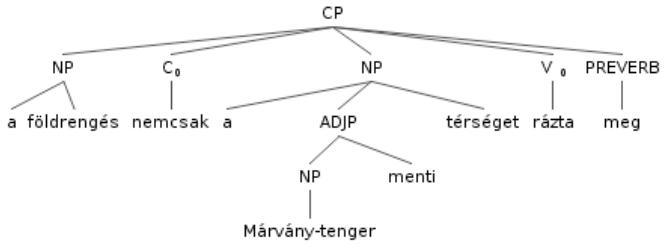
Az adatok kinyerésekor feljegyezzük azt is, hogy az adott NP-be milyen mélyen ágyazódnak további NP-k, így lehetőségünk nyílik egyfajta komplexitásfogalom alapján több chunk-típust megkülönböztetni. Így például az (1) alatti frázisok, amennyiben maximális NP-t alkotnak, rendre `N_1`, `N_2`, `N_3` típusú chunknak címkézzük.

(1a) $[_{NP}$Az elnök]
(1b) $[_{NP}[_{NP}$Az Egyesült Államok$][_{NP}$elnöke]]

(1c) $[_{NP}[_{NP}[_{NP}$Az Egyesült Államok$][_{NP}$elnökének$]]$ irodája$]]$

Ezen információ kinyerését nem tekintjük a címkéző feladatának, csupán a gépi tanulási feladatot könnyítjük meg vele: az NP-felismerés pontosságának javulását várjuk attól, hogy a modellnek lehetősége van olyan szabályszerűségeket is tárolni, melyek csak bizonyos „mélységű” NP-kre jellemzőek. Végül a legjobb eredményeket azzal a címkézéssel értük el, ahol csupán a legalacsonyabb – tehát további NP-t nem tartalmazó – főnévi csoportokat különböztettük meg (`N_1`) a komplexebbektől (ezeket `N_2+`-szal jelöltük).

A fenti definíció és cimkézés eredményeképp a Szeged Treebank 1. ábrán látható mondata az NP-korpuszban az 2. ábrán látható módon jelenik meg.



1. ábra. A Szeged Treebank egy mondatának elemzése

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

2. ábra. A Szeged Treebank egy mondatának chunk-címkézése

3. A `hunchunk` rendszer

3.1. Felügyelt tanulás

Az NP-felismerő eszközök többsége *felügyelt tanulásra* épül. Ez a kifejezés azt jelenti, hogy a számítógéppel elvégezni kívánt feladathoz (általában emberi munkával) *tanulóadathalmazt* hozunk létre, amely nagyszámú bemenet-kimenet párból áll. Egy felügyelt tanulóalgoritmus feladata, hogy a tanulóadat alapján automatikusan tárjon fel összefüggéseket a bemenet és kimenet között, és az így létrehozott *modell* segítségével később új, korábban nem látott bemenetre meghatározza a legvalószínűbb kimenetet. *Címkézésről* beszélünk, ha a kimenet egy rögzített véges halmazból kerül ki.

Szekvenciális címkézési feladatról beszélünk, ha a feladat struktúrája olyan, hogy a bemenet egy sorozat (pl. egy mondat tokenjeinek sorozata), a kimenet

pedig a sorozat minden eleméhez címkét rendel. Felhívjuk a figyelmet arra, hogy a szekvenciális címkézés szigorú értelemben nem alesete a címkézésnek, hiszen kimenetének hossza függ a bemenet hosszától. Az NP-felismerésen kívül számos más nyelvtechnológiai feladat is megoldható felügyelt szekvenciális címkézési módszerekkel, így például a szófaji címkézés (part-of-speech tagging, POS-tagging) és a tulajdonnévfelismerés (named entity recognition, NER).

A felügyelt tanulás feladatát megoldó matematikai módszereknek a bemenetet valamilyen strukturált formában kell megkapniuk. Ennek egy standard módja, hogy a bemenetet (vagy szekvenciális címkézésnél annak egyes tokenjeit) ún. *jegyek* halmazával írjuk le; ilyen jegyre lehetséges példa az, hogy a token nagy betűvel kezdődik-e, ige-e, többes számban áll-e, stb. A tanulóalgoritmus ezek alapján modellépítéskor olyan statisztikai összefüggéseket tár fel, mint például hogy a token nagy betűvel kezdődése hogyan befolyásolja annak az esélyét, hogy a token a `B-N_1` címkét kapja. Címkézéskor az újonnan érkezett bemenet jegyei alapján megállapítja, hogy mi a modell által legvalószínűbbnek ítélt címkézés. Noha az itt bemutatott rendszer kizárólag bináris jegyeket használ – azaz egy jegynek csak két értéke lehet, például egy szó vagy nagybetűs, vagy nem –, a tanulóalgoritmus lehetővé teszi, hogy egy-egy jegy tetszőleges valós értéket felvegyen.

A felügyelt tanítás módszere mögött ott van az alapfeltevés, hogy a tanítóadathalmazon feltárt összefüggések relevánsak lesznek az új adatok feldolgozásakor is. Ez az alapfeltevés csak akkor tekinthető kellően megalapozottnak, ha az új szöveg jellege (zsánere) nem tér el nagyon a tanulóadatétól. Ugyanakkor megállapítható, hogy a rendszerünk által automatikusan feltárt milliónyi statisztikai összefüggés közül a statisztikai értelemben legerősebbek zsánertől független, általános jelenségeket számszerűsítenek.

A fejezet hátralévő részében részletesen bemutatjuk, hogy rendszerünk számára hogyan reprezentáljuk jegyekkel a bemenetet. Az ezután következő fejezet ismerteti magát a tanuló- illetve címkézőalgoritmust.

3.2. Jegyek

Egy token legfontosabb jegyei a szóalak és annak valamennyi morfológiai jegye. A szóalak jegyként felvétele szükséges ahhoz, hogy a tanulóalgoritmusnak lehetősége legyen olyan statisztikai összefüggések megtalálására, mint például: „a *ha* szó ritkán áll NP belsejében”. Egy szó morfológiai jegyeinek reprezentációjára számos különböző konvenció létezik. A Szeged Treebank az MSD-kódolást követi (Erjavec 2004), ezt az NP-korpusz létrehozásakor azonban átalakítottuk az ún. KR-formalizmusra (Rebrus et al. 2004), mivel az általunk használt morfológiai cimkéző, elemző és egyértelműsítő egyaránt ezt a formátumot követi. A KR-formalizmus előnye, hogy egy szó valamennyi morfológiai jegyét külön karaktersorozatnak felelteti meg, így a KR-kódok jelentése kompozicionális (például az *asztalát* szó a `NOUN<POSS><CAS<ACC>>` kódot kapja). A KR-konvencióra való áttérés lehetővé tette, hogy ezen kódok valamennyi, egy-egy morfológiai jegynek megfeleltethető összetevőjét jegyként vegyük fel. Az eddig bevezetett, egy-egy szóra a többitől függetlenül kiértékelhető jegyeket a **??**. táblázat foglalja össze.

jegytípus	jegyek
szóalak	`mesélte`
n-gram	`mes`, `esé`, `sél`, `élt`, `lte`
KR	`VERB`, `PAST`, `DEF`

1. táblázat. A *mesélte* szó jegyei

Fontos megemlítenünk, hogy míg a tanulókorpuszban a morfológiai jegyek készen rendelkezésre állnak, addig nem látott szöveg címkézéskor más a helyzet. Hogy korábban nem látott mondatok NP-chunkolását is elvégezhesse a rendszer, a nyers szöveg szavaihoz előfeldolgozási lépésként morfológiai címkéket kell rendelnünk. A `hunpos` morfológiai címkéző (Halácsy et al. 2007), maga is felügyelt címkézési módszereket alkalmazva, megoldja ezt a feladatot.

Mivel feltételezzük, hogy egy szó címkéjének valószínűségét a környezetében található szavak tulajdonságai is befolyásolják, ezért a jegyeket minden tokenre annak 5 szavas környezetében értékeljük ki, tehát egy-egy token jegyei közt szerepel az önmagára, valamint az őt megelőző és követő 5-5 tokenre vonatkozó információ is. A `-1_form=Az` jegy például azt jelenti, hogy az ezzel a jeggyel rendelkező tokent megelőző szó alakja *Az*, a `2_kr=PLUR` jegy azt jelenti, hogy a kettővel utána következő szó többesszámú, stb.

Bevezettük továbbá az ún. *szófajminta-jegyet*, mely egy szó adott hosszúságú környezetében az egymást követő szavak szófaji cimkéinek sorozatait írja le a következő módon: ha a jegy sugarát r-el, 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. Más szavakkal minden szóhoz feljegyezzük, hogy valamekkora hosszúságú környezetében milyen szófajú (kategóriájú) szavak sorozatait találjuk. Minden jegyérték elején feltüntetünk két számot, melyek azt jelölik, hogy a szóbanforgó tokenhez képest az adott szófaj-sorozat hol helyezkedik el. Így a 2. táblázatban szereplő mondat *csapos* szava a 3. táblázatban szereplő jegyeket veszi fel a szófajminta jegy értékeként.

A	csapos	mesélte	,	hogy	milyen	szép	kés	van	bennem	.
`ART`	`NOUN`	`VERB`	`PUNCT`	`CONJ`	`ADJ`	`ADJ`	`NOUN`	`VERB`	`NOUN`	`PUNCT`

2. táblázat. A *Szeged Treebank* egy mondatának szófaji címkézése

A KR-mintákat kiválasztó jegy sugarát növelve a chunkolás minősége is nő, 3-nál magasabb sugár mellett azonban a jegyek túl magas száma nem teszi lehetővé a modell tanítását.

Az alábbiakban bemutatjuk a modellezési eljárást, amellyel a szekvenciális címkézési feladatot megoldjuk. A matematikai részletekben legkevésbé sem elmerülve inkább arra törekszünk, hogy a statisztikai módszerekben nem feltétlenül járatos olvasó képet kapjon arról, hogy milyen heurisztikák vezérlik

-1_1_ART+NOUN
-1_2_ART+NOUN+VERB
-1_3_ART+NOUN+VERB+PUNCT
-1_4_ART+NOUN+VERB+PUNCT+CONJ
0_2_NOUN+VERB
0_3_NOUN+VERB+PUNCT
0_4_NOUN+VERB+PUNCT+CONJ
1_3_VERB+PUNCT
1_4_VERB+PUNCT+CONJ
2_4_PUNCT+CONJ

3. táblázat. Egy token szófajminta-jegyei ($r = 4$)

egy a miénkhez hasonló statisztikai modell megalkotását, és milyen standard technikákra támaszkodhat a modellező kutató. Módszerünk részletesebb leírása megtalálható (Recski et al. 2009).

3.3. Maximum Entrópia modell

A jegyek és címkék közötti összefüggések feltárásához a Maximum Entrópia módszert (MaxEnt) választottuk (Ratnaparkhi 1998). Ezen eljárást már sikerrel alkalmaztuk összetevőként többek között morfológiai címkézés (Halácsy et al. 2005) és tulajdonnévfelismerés (Varga–Simon 2007) feladatának megoldásakor. A módszer fontos tulajdonsága, hogy felhasználásakor egy-egy szóra – annak jegyei alapján – nem csupán a legvalószínűbb címkét adja meg, hanem valamennyi lehetséges címkéhez valószínűségeket rendel; a modell ezen képessége elengedhetetlen ahhoz, hogy egy mondat legvalószínűbb címkézését megtaláljuk (ez utóbbi folyamatot a későbbiekben ismertetjük majd).

A Maximum Entrópia módszer pontos ismertetésére itt nem nyílik lehetőségünk, de egy példán keresztül mutatjuk be alkalmazásunk szempontjából legfontosabb sajátosságait. Tegyük fel, hogy a tanulókorpuszt vizsgálva megállapítjuk, hogy a `kr=PLUR` jeggyel rendelkező tokenek körében kétszer akkora az `E-N_1` címke esélye, mint az ilyen jegyet nem kapott tokenek körében. Ekkor első közelítésben kiindulhatunk abból, hogy a jegy jelenléte kétszeresére növeli a `E-N_1` címke esélyét a nem látott adatokon is. Ezt úgy mondjuk, hogy a `kr=PLUR` jegy jelenlétére nézve a `E-N_1` címkének kettő az *esélyhányadosa* (odds ratio). Ha több jegy áll rendelkezésünkre a döntéshez, akkor az egyes jegyekhez tartozó esélyhányadosokat összeszorozva kiszámolhatjuk az egyes címkék egymáshoz viszonyított esélyeit. Így működik az ún. Naív Bayes módszer, ahol a modell egyszerűen ezeknek az esélyhányadosoknak a tanulókorpusz alapján épített táblázata. A Naív Bayes módszer nyilvánvaló hibája, hogy figyelmen kívül hagyja, hogy a jegyek által adott esélyhányadosok valójában függenek attól, hogy milyen más jegyek vannak jelen. Például a `kr=PLUR` jegyből kinyerhető információ redundáns akkor, ha már kinyertük a `form=azok` jegyből megszerezhető információt. Ezt a problémát orvosolja a Maximum Entrópia módszer. Egy MaxEnt modell ugyanúgy a jegyekhez rendelt esélyhányadosok

táblázata, mint egy Naív Bayes modell, de olyan módon határozza meg az esélyhányadosokat, hogy az ilyen kettős számlálásokat lehetőség szerint kiküszöbölje.

3.4. Átmenetmodell

Egy szekvenciális címkézési feladat megoldásának egy egyszerű, kevéssé kifinomult módja, hogy eltekintünk a szekvencialitástól: olyan modellt tanítunk, amely minden egyes tokenre külön-külön hozza meg a döntését, a token jegyei alapján, például az előző alfejezetben bemutatott MaxEnt módszerrel. Már egy ilyen modell is képes jó minőségű chunkolásra (lásd a 4.2. alfejezetet), de nem használja ki explicit módon a tokenek címkéi közötti összefüggéseket. Ennek következtében például az is előfordulhat, hogy a kimenetben egy `I-NP` címkét `B-NP`, egy `O` címkét `E-NP`, stb. követ, pedig ezek szabálytalan, chunkolásként értelmezhetetlen címkesorozatok, amelyek a tanulókorpuszban nem is fordultak elő. A szabálytalan szekvenciák kiszűrésén túl azt is figyelembe szeretnénk venni, hogy a szabályos címkesorozatok sem egyformán gyakoriak: Az `E-NP` címkét például csak akkor követi `B-NP` vagy `1-NP`, ha egy NP-t közvetlenül követ egy másik – ez valószínűtlenebb, mint az `E-NP O` szekvencia, amikor egy NP-t NP-n kívüli szó követ, például egy ige.

A címkék közötti korrelációk figyelembevételének talán legegyszerűbb módja az *átmenetmodell*, amely csak a szomszédos címkék közti összefüggést modellezi, pontosabban azt becsli, hogy egy adott címke után milyen eséllyel következik egy adott másik. Ennek megépítéséhez a tanítókorpuszban megszámoljuk valamennyi, két címke hosszúságú sorozatot (*bigram*ot), majd minden címkére feljegyezzük, hogy azon bigramok közül, melyeknek ő az első tagja, milyen arányban szerepelnek a különböző címkék a második helyen.

Az átmenetmodell segítségével egyszerű szorzat formájában írhatjuk fel egy adott címkesorozat megfigyelésének valószínűségét:

$$P(t_1 t_2 \dots t_n) = P(t_n|t_{n-1})P(t_{n-1}|t_{n-2}) \dots P(t_2|t_1)P(t_1)$$

Ez a képlet várakozásainknak megfelelően nulla valószínűséget rendel a szabálytalan címkesorozatokhoz.

3.5. Címkézés

Ahogy az előző két alfejezetben bemutattuk, a tanítókorpusz alapján két modellt építettünk, a jegyek és egyetlen címke kapcsolatát feltáró Maximum Entrópia modellt, illetve a a címkék egymáshoz való viszonyáról nyilatkozó átmenetmodellt. Ebben az alfejezetben azt tárgyaljuk, hogy egy új, a tanulókorpuszban feltehetőleg nem szereplő mondathoz hogyan találhatjuk meg a fenti két modell ismeretében legvalószínűbb címkézést. Az alább ismertetett módszert elsőként (McCallum et al. 2000) alkalmazta címkézési feladatok megoldására.

A két modell kombinálásához – az itt most nem tárgyalt Rejtett Markov Folyamatok (Hidden Markov Model, HMM, ld. pl.: Rabiner 1989) elmélete által motivált módon – egy leegyszerűsítő feltételezéssel élünk: azt tételezzük fel, hogy az az információ, amit egy címkéről az előző címke ad, független attól az információtól, amit a jegyek alapján a MaxEnt modell ad. Ennek a feltételezésnek a matematikailag pontos megfogalmazásához észre kell vennünk, hogy a függetlenség valószínűségszámítási értelemben nem teljesül, amennyiben mindkét modell előnyben részesíti a gyakoribb címkéket. Ez a megfontolás vezet az alábbi képlet végső formájához, amely (fix, csak w-től függő szorzótényezőtől eltekintve) megadja, hogy egy t címkesorozat mennyire teszi valószínűvé egy w mondat megfigyelését.

$$P(w|t) \sim \prod_i \frac{P(t_i|w)P(t_i|t_{i-1})}{P(t_i)}.$$

A valószínűséget ($P(w|t)$) tehát úgy kapjuk meg, ha minden címke esetében összeszorozzuk a MaxEnt modell által szolgáltatott valószínűséget ($P(t_i|w)$) az átmenetvalószínűséggel ($P(t_i|t_{i-1})$), korrigálva a címke relatív gyakoriságával ($P(t_i)$). A képlet levezetését lásd (Recski et al. 2009).

A fenti képlet egy adott címkézés valószínűségét értékeli ki, de a mi feladatunk nem ez, hanem az összes lehetséges címkézés közül a legvalószínűbbet megtalálni. Az elvben lehetséges címkézések rendkívül nagy száma miatt az egyenkénti kiértékelésük nem lenne kivitelezhető. A fenti képlet speciális formája miatt azonban létezik hatékony algoritmus annak a címkézésnek a megtalálására, amely a képletet maximalizálja. Ez a Viterbi algoritmus, amelynek részleteit itt szintén nem ismertetjük, de lényege, hogy a mondat szavain balról jobbra végighaladva a mondat minden kezdőszeletéhez és minden lehetséges címkéhez meghatározza, hogy mi a kezdőszelet legvalószínűbb címkézése azok közül, amelyek az adott címkével végződnek.

A címkézőeszközünknek szabadon változtatható paramétere, hogy a címkemodellt – azaz az egyes chunk-címkék közötti átmenetvalószínűségeket tartalmazó modellt – a MaxEnt modellhez képest milyen súllyal vegye figyelembe (ez bevett eljárás a Rejtett Markov Modellezés területén). A fenti képletet ez úgy általánosítja, hogy a $P(t_i|t_{i-1})$ és a $P(t_i)$ kifejezéseket valamely pozitív λ kitevővel hatványozzuk. A λ paraméter legjobb eredményt biztosító értékét úgy kerestük meg, hogy a tanítókorpusz egy kisméretű részén megmértük, mely beállítás adja a legjobb eredményt.

4. Kiértékelés

4.1. Módszertan

NP-felismerőnk kiértékeléséhez a korpusz tanításra fel nem használt, kb. százezer tokenből álló részét használtuk fel. A tesztkorpuszon lefolytatott cimkézések kimenetét a CoNLL 2000 feladat szigorú előírásait követve értékeltük ki, tehát akkor és csak akkor tekintettünk egy főnévi csoportot helyesen azonosítottnak,

ha az eszközünk az NP minden tokenjét – és csak azokat – ismerte fel NP részeként.

Az eszköz teljesítményét – a szakirodalomban megszokott módon – mind *pontosság*, mind *fedés*, mind pedig a kettőből kiszámítható *F-pontszám* segítségével jellemezzük. A pontosság azt mutatja meg, hogy a címkéző által azonosított NP-k hány százaléka helyes, a fedés ezzel szemben annak mérőszáma, hogy a tényleges NP-k közül mennyit találtunk meg. A legtöbb osztályozási feladat során könnyen lehet e két értékből valamelyiket a másik rovására magasan tartani, ezért szokásos az ilyen eszközök teljesítményét az F-pontszámmal jellemezni, mely a pontosság és fedés harmonikus közepeként áll elő. (A harmonikus közép azt az elvárásrendszert formalizálja, amikor a tévesen NP-ként azonosított szövegrészek (false positive) ugyanolyan súlyú hibának tekintettek, mint az észre nem vett NP-k (false negative)).

4.2. Eredmények

Az osztályozási feladatokon elért eredményeket szokásos eljárás szerint egy ún. *baseline* módszerhez hasonlítjuk, mely általában valamely egyszerű heurisztikát jelent, melyhez képest jobb teljesítmény elérése a bemutatott módszer minimális célkitűzése. Az általunk választott baseline módszer lényege, hogy az egyszerűbb IOB címkézési konvenciót követve (ld. a 2 fejezetet) minden szóhoz azt a címkét rendeljük, amely a szó kategóriája (szófaja) alapján a legvalószínűbb, tehát amely címke az adott kategóriájú szavak mellett a leggyakrabban figyelhető meg a tanítókorpuszban. E módszer, valamint a `hunchunk` rendszer eredménye a 4. táblázatban látható. Tájékoztatásul feltüntetjük a címkék közti átmenetvalószínűségeket figyelmen kívül hagyó, csupán a MaxEnt modell kimenetére támaszkodó rendszer eredményét is. Végül a táblázat azon rendszer teljesítményét is megmutatja, melynek a tesztkorpusz kézzel készült morfológiai elemzése nem állt rendelkezésre, csupán a `hunmorph` morfológiai elemző kimenetére támaszkodhatott.

	Pontosság	Fedés	F-pontszám
baseline	60.24%	60.50%	60.37%
csak MaxEnt	88.32%	87.54%	87.93%
hunchunk	**90.58%**	**89.98%**	**90.28%**
hunchunk+hunpos	87.27%	86.32%	86.79%

4. táblázat. A `hunchunk` rendszer teljesítménye

4.3. A CoNLL feladat

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 az alap NP-k. Ez magyarázza

a szakirodalomban szokásosan láthatónál alacsonyabb pontszámokat. Noha figyelmünk középpontjában a maximális NP-k azonosítása volt, algoritmusunk teljesítményét a state-of-the-art statisztikai szegmentálóalgoritmusokéval is össze kívántuk vetni, ezért a már említett angol nyelvű CoNLL 2000 feladaton is kipróbáltuk. A CoNLL 2000 feladat tanuló- és tesztadata rögzített, ezáltal szolgálhat a különböző szegmentálóalgoritmusok összehasonlításának standard terepeként. Eszközünk 93.79%-os F-pontszámot ért el a feladaton, míg a legmagasabb publikált eredmények között szerepel például 94.34% (Sun et al. 2008) és 94.29% (Sha–Pereira 2003). Bár ez utóbbi eredményektől rendszerünk kb. fél százalékponttal elmarad, fontosnak tartjuk megemlíteni, hogy azoknak a komplexebb modelleknek a tanítása, melynek segítségével ezek az eredmények születtek (Conditional Random Field, CRF, cf. Lafferty et al. 2001) akár egy nagyságrenddel hosszabb időt vesz igénybe, mint az általunk bemutatott modellezési eljárás.

5. További tervek

A 2.1. fejezetben már megemlítettük, hogy az általunk választott feladatnak – a maximális NP-k azonosításának – egyik célja, hogy a későbbiekben lehetőségünk nyíljon mondatok közvetlen összetevőinek, elsősorban az igék vonzatainak feltérképezésére, melynek hasznát vehetjük gépi fordítási feladat során. Ennek első lépéseként szükséges elkészítenünk egy magyar-angol párhuzamos, azaz ugyanazon szövegeket két nyelven tartalmazó NP-korpuszt, melyhez kiindulópontként szolgált a Hunglish magyar-angol párhuzamos korpusz (Varga et al. 2005). Megfelelő tanulóadat előállítását követően a `hunchunk` eszközt alkalmassá tettük angol nyelvű NP-felismerésre is (bővebben ld. Recski et al. 2009), így elvégezhettük a Hunglish korpusz NP-chunkolását mindkét nyelven. További terveink között szerepel a korpusz NP-szintű párhuzamosítása – azaz a megfelelő NP-k összerendelése –, később egy felügyelt tanulásra épülő magyar-angol NP-fordító létrehozása. Elképzeléseink szerint az NP-fordítás képességével jelentősen közelebb kerülnénk egy jó minőségű angol-magyar fordítórendszer létrehozásához is. A bemutatott rendszert továbbá – megfelelő új tanítókorpusz létrehozásával – alkalmassá tettük tetszőleges kategóriájú maximális mondattani összetevők azonosítására (Recski 2011), ezzel újabb lépést téve a magyar mondatok felszíni szerkezetének gépi azonosítása felé.

Hivatkozások

1. Steven P. Abney. Parsing by chunks. *Bell Communications Research*, 1991.
2. Dóra Csendes, János Csirik, Tibor Gyimóthy, and András Kocsor. The Szeged Treebank. In *Lecture Notes in Computer Science: Text, Speech and Dialogue*, pages 123–131, 2005.
3. Tomaž Erjavec. MULTEXT-East version 3: Multilingual morphosyntactic specifications, lexicons and corpora. In Nicoletta Calzolari, editor, *Fourth International Conference on Language Resources and Evaluation, (LREC'04)*, volume 4, pages 1535–1538. ELRA, 2004.

4. James Paul Gee and François Grosjean. Performance structures: A psycholinguistic and linguistic appraisal. *Cognitive Psychology 15*, pages 411–458, 1983.
5. Péter Halácsy, András Kornai, and Csaba Oravecz. HunPos – an open source trigram tagger. In John A. Carroll, Antal van den Bosch, and Annie Zaenen, editors, *Annual Meeting of the Association for Computational Linguistics*, volume 45, 2007.
6. Péter Halácsy András Kornai, and Dániel Varga. Morfológiai egyértelműsítés Maximum Entrópia módszerrel. In Zoltán Alexin and Dóra Csendes, editors, *III. Magyar Számítógépes Nyelvészeti Konferencia*, pages 180–189. Szegedi Tudományegyetem, 2005.
7. John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Carla E. Brodley and Andrea Pohoreckyj Danyluk, editors, *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*, pages 282–289. Morgan Kaufmann, 2001.
8. Mitchell P. Marcus, Beatrice Santorini, and Mary A. Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19:313–330, 1994.
9. Márton Miháltz. Magyar NP-felismerők összehasonlítása. In Attila Tanács and Veronika Vincze, editors, *VIII. Magyar Számítógépes Nyelvészeti Konferencia*, pages 333–335, 2011.
10. Gábor Prószéky, László Tihanyi, and Gábor Ugray. Moose: A robust high-performance parser and generator. In John Hutchins, editor, *Proceedings of the 9th EAMT Conference*, pages 138–142. Foundation for International Studies, 2004.
11. R. Lawrence Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. In *Proceedings of the IEEE*, volume 77, pages 257–286, 1989.
12. Lance A. Ramshaw and Mitchell P. Marcus. Text chunking using transformation-based learning. In David Yarowsky and Kenneth Church, editors, *Proceedings of the Third Workshop on Very Large Corpora*, pages 82–94. Massachusetts Institute of Technology, 1995.
13. Adwait Ratnaparkhi. *Maximum Entropy Models for Natural Language Ambiguity Resolution*. PhD thesis, University of Pennsylvania, 1998.
14. Péter Rebrus, Péter Vajda, Péter Halácsy, András Rung, and Viktor Trón. Általános célú morfológiai elemző kimeneti formalizmusa. *II. Magyar Számítógépes Nyelvészeti Konferencia*, 2004.
15. Gábor Recski. A sekély mondattani elemzés további lépései. In Attila Tanács and Veronika Vincze, editors, *VIII. Magyar Számítógépes Nyelvészeti Konferencia*, pages 113–118, 2011.
16. Gábor Recski, Dániel Varga, Attila Zséder, and András Kornai. Főnévi csoportok azonosítása magyar-angol párhuzamos korpuszban. In Attila Tanács, Dóra Szauter, and Veronika Vincze, editors, *VI. Magyar Számítógépes Nyelvészeti Konferencia*, pages 3–13, 2009.
17. Erik F. Tjong Kim Sang and Sabine Buchholz. Introduction to the conll shared task: Chunking. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL 2000 and LLL 2000*, pages 127–132. Association for Computational Linguistics, 2000.
18. Erik F. Tjong Kim Sang and Jorn Veenstra. Representing text chunks. In Henry S. Thompson and Alex Lascarides, editors, *Proceedings of the Ninth conference of the European chapter of the Association for Computational Linguistics (EACL '99)*, pages 173–179. Association for Computational Linguistics, 1999.

19. Fei Sha and Fernando Pereira. 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*, pages 134–141, Morristown, NJ, USA, 2003. Association for Computational Linguistics.
20. Xu Sun, Louis-Philippe Morency, Daisuke Okanohara, and Jun'ichi Tsujii. 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*, pages 841–848, Morristown, NJ, USA, 2008. Association for Computational Linguistics.
21. Kiyotaka Uchimoto, Qing Ma, Masaki Murata, Hiromi Ozaku, and Hitoshi Isahara. Named entity extraction based on a maximum entropy model and transformation rules. In Hitoshi Iida, editor, *ACL '00: Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 326–335, Morristown, NJ, USA, 2000. Association for Computational Linguistics.
22. Tamás Váradi. Shallow parsing of Hungarian business news. In Dawn Archer, Paul Rayson, Andrew Wilson, and Tony McEnery, editors, *Proceedings of the Corpus Linguistics 2003 Conference*, pages 845–851. UCREL, 2003.
23. Dániel Varga, László Németh, Péter Halácsy, András Kornai, Viktor Trón, and Viktor Nagy. Parallel corpora for medium density languages. In Nicolas Nicolov, Kalina Bontcheva, Galia Angelova, and Ruslan Mitkov, editors, *Recent Advances in Natural Language Processing IV: Selected Papers from RANLP 2005*, pages 247–258. John Benjamins Publishing Company, 2005.
24. Dániel Varga and Eszter Simon. Hungarian named entity recognition with a Maximum Entropy approach. *Acta Cybernetica*, 16:293–301, 2007.