XVIII. Magyar Számítógépes Nyelvészeti Konferencia Szeged, 2022. január 27–28.

Igekötő-kapcsolás

Pethő Gergely, Sass Bálint, Kalivoda Ágnes, Simon László, Lipp Veronika

Nyelvtudományi Kutatóközpont
pagstudium@gmail.com
{sass.balint,kalivoda.agnes,simon.laszlo,lipp.veronika}@nytud.hu

Kivonat Ahogy ebben a mondatban is látjuk, a magyarban az igekötő el tud válni az igéjétől. A korpuszok alapvető egysége a szó, a token, emiatt a magyar nyelvű korpuszokban hagyományosan mindig külön tokenként jelenik meg az elvált igekötő. Azt az alapvető információt tehát, hogy melyik igéhez tartozik az adott igekötő, ezek a korpuszok nem tartalmazzák. Jelen tanulmányban egyrészt (1) módszert adunk az igekötők alapigéhez kapcsolására, másrészt (2) egy sémát arra, hogy ezt hogyan jelenítsük meg a korpusz annotációjában a korpuszban való keresést leginkább támogató módon. Az eszközt az `e-magyar` rendszer moduljaként implementáltuk, olyan funkcionalitást valósít meg, amelyre számos kutatásban merül fel igény. Az elkészült `emPreverb` modul és a kézzel annotált fejlesztő és tesztkorpuszok szabadon felhasználhatók.
Kulcsszavak: igekötő, elváló igekötő, alapige, keresés korpuszban, korpusz, korpusznyelvészet, lexikográfia

1. Be#vezetés

Korpuszaink alapstruktúrája a legtöbb esetben: szavak sorozata. Ez nagyon leegyszerűsítve szóközökkel elválasztott karaktersorozatok sorozatát jelenti. A tokenizálás művelete ezt sok ponton finomítja. A szavakat, írásjeleket, azaz a korpusznak a tokenizáló által szolgáltatott alapegységeit nevezzük tokeneknek.

Amit a tokenizáló sosem szokott megcsinálni, hogy egy tokenként értelmezze két olyan elem együttesét, melyek közé közbeékelődik egy további elem. Ennek következtében tokenizálás után az elvált igekötő külön token lesz, azaz a tapadó igekötős ige egy token lesz, az elvált igekötő és az igéje viszont kettő. Mondhatjuk, hogy ez nem a tokenizáló feladata, ezt majd megoldják a magasabb elemzési szintek, de attól még a tény tény marad, hogy egy morfológiailag elemzett korpuszban nagyságrenddel nehezebb lesz megkeresni egy igekötős ige összes alakját, mint mondjuk egy főnévét.

Valóban, érdemes úgy tekinteni a dolgot, hogy a '*megkeres*' és a '*keres meg*' ugyanannak a lexémának két alakja, azaz gyakorlati szempontból – lexikográfiai és korpusznyelvészeti megfontolásokat szem előtt tartva – a '*keres meg*' is „egy szó". Azontúl, hogy „van benne szóköz", nem is nagyon van jó érv amellett, hogy ez két lexéma lenne.

A magyar szótárakban az igekötős igék hagyományosan saját szócikket kapnak. Ezek a szócikkek egyaránt szólnak a tapadó és az elvált alakokról, egyaránt

tartalmaznak olyan példaszövegeket, amelyekben az igekötő elvált az igéjétől, meg olyanokat, amelyekben nem: mindkét esetre ugyanannak a lexémának az adataként tekintünk. *A magyar nyelv nagyszótára* (Ittzés és mtsai, 2006–2021) eddig elkészült köteteiben a címszavak 14%-a igekötős ige. Ha pedig az igekötős igék korpusznyelvészeti vizsgálatáról van szó, akkor is beleértik mindkét alaktípust, sokszor éppen ezek eloszlása, használata, fajtái a kutatás tárgya.

Azokban a korpuszvizsgálatokban, amikor nemcsak példákat keresünk, hanem átfogóan, statisztikailag akarunk elemezni egy jelenséget, az alapvető kiindulópont az összes releváns korpuszhelyet tartalmazó találati lista, illetve az ebből készített gyakorisági lista. Az '*elmegy*' gyakorisági listáján nyilvánvalóan a '*megy el*', '*el sem fog tudni menni*' stb. találatoknak is szerepelniük kell.

Az általunk ismert magyar nyelvű, elemzett korpuszok ennek a követelménynek nem felelnek meg. Bár a probléma az elmúlt évek számos olyan lexikográfiai (Lipp és Simon, 2021) és nyelvészeti projektje kapcsán előkerült, amely a korpuszalapú vagy korpuszvezérelt megközelítést vallotta magáénak, eddig nem született rá átfogó megoldás. Ebben a tanulmányban ezt a megoldást keressük két aspektusból: bemutatunk (1) egy *igei természetű szavak* (igék, igenevek és deverbális főnevek) és hozzájuk tartozó igekötők összerendelésére szolgáló eszközt és (2) egy sémát arra, hogy hogyan érdemes ezt a korpuszban reprezentálni úgy, hogy az igekötős szavakat érintő korpuszalapú munkát minél kényelmesebbé, könnyebbé tegyük. Azt reméljük, hogy eredményeinket széles körben lehet majd hasznosítani. A továbbiakban az igekötőket a `[/Prev]`, az igéket a `[/V]` kóddal fogjuk jelölni szükség esetén.

2. Kapcsolódó munkák

2.1. Igekötő-kapcsolás

A magyar nyelvű korpuszokat nézve: az MNSZ-ben (Oravecz és mtsai, 2014) és a TMK-ban (Dömötör és mtsai, 2017) nincsen igekötő-kapcsolás, az Ómagyar korpuszban (Simon és Sass, 2012) – mely a kezdetektől a majdani szintaktikai elemzés igényével készült – viszont van, a feladatot itt manuális munkával oldották meg.

Az első automatikus, szabályalapú módszert a Mazsola adatbázis készítéséhez fejlesztették (Sass, 2011, 38–39. oldal), ez egyszerűen a morfológiai kódok alapján kapcsolja össze az igekötőt és az igét, és azt sem ellenőrzi, hogy volt-e eredetileg igekötő az igealakon. Tudomásunk szerint a Mazsola adatbázisa egyben az egyetlen kutatási célra elérhető nagyméretű magyar nyelvű korpusz, melyben automatikus igekötő-kapcsolás van. Természetesen a Szeged Treebank (Csendes és mtsai, 2005) is tartalmaz – manuális munkával készített – igekötő-kapcsolást. A második módszer (Recski, 2011, 3.1 rész) szintén szabályalapon működik, négy viszonylag triviális szabály alkalmazásával 0,964-es F_1-pontszámról számol be. Ez a módszer morfológiailag elemzett korpuszon dolgozik, az eredménye révén a szintaktikai elemzéshez kíván hozzájárulni. Ugyancsak szabályalapú módszert alkalmaz a VFrame eljárás (Vadász és mtsai, 2017, 2018), amely az igekötő-

kapcsolásnak egy részproblémáját oldja meg: a finit igéhez, valamint az infinitívuszhoz tartozó igekötők megtalálását az ige szűk kontextusában. Novák és mtsai (2019) részfeladatként utalnak a témára, megjegyzik, hogy az elvált esetben az igék lemmája nem tartalmazta az igekötőt, „ezért ezt a hibát is kijavították," de a hogyanról nem nyilatkoznak. Megemlítjük, hogy az eredeti `e-magyar` rendszerben (Sass és mtsai, 2017) létezett egy segédmodul, ami a szintaktikai elemzés alapján kapcsolja az igéhez az igekötőt. Így elindulva a feladat nyilván triviális.

A németben a magyarhoz hasonlóan létezik elváló igekötő, azaz felmerül a szóban forgó feladat, és az utóbbi időben többen foglalkoztak is vele (Batinić és Schmidt, 2018; Köper, 2018). Volk és mtsai (2016) célkitűzése hasonló a jelen tanulmányéhoz, módszere triviálisabb: egyszerűen az igekötőhöz *balra* legközelebbi finit igéhez kapcsolják az igekötőt, ha az igekötős ige benne van a szótárban. A németben marginális jelentőségű topikalizált igei partikulát (tehát ami az igétől balra esik) az egyszerűség kedvéért nem is igei partikulának, hanem határozószónak tekintik; ezek a magyar nyelv esetében nem intézhetők el ilyen könnyen.

2.2. Reprezentáció a korpuszban

Az MNSZ (Oravecz és mtsai, 2014) csupán elkülöníti az igekötőtlen igéket az igekötősektől (`IGE` vs `IK.IGE`), a TMK-ban (Dömötör és mtsai, 2017) viszont a tapadás a morfológiai kódon túl az ige lemmájában is jelölődik. Az elvált igekötő hovatartozásáról egyik sem ad információt.

Az Ómagyar korpuszban (Simon és Sass, 2012) viszont már megjelenik ez: az alapigénél külön attribútumként szerepel a hozzá tartozó elvált igekötő lemmája. Ezt az információt meglehetősen nehéz előhívni a keresés során, de a törekvésből látszik, hogy mennyire fontos ügy ez – szintaktikai és magasabb szintű elemzésekhez mindenképpen.

A Mazsolában (Sass, 2009) az elvált igekötő az alapige lemmája elé van kapcsolva, ez nagy előrelépés, ugyanakkor ebben a reprezentációban az elválás/tapadás információ elvész: a '*keres meg*' is egyszerűen '*megkeres*'-ként jelenik meg. Novák és mtsai (2019) is így jár el, azaz az igekötőt *beleírja* az igelemmába.

Az eredeti `e-magyar` rendszer (Sass és mtsai, 2017) kicsit szofisztikáltabb, de nem feltétlenül jobb megközelítéssel él. Szintén „az igekötő és az igealak szótövének egybeírásaként az igekötős szótövet adja meg elvált esetben is", de az így létrejött lemmát egy külön attribútumban tárolja (`lemmaWithPreverb`) a lemmától elkülönítve, ami a keresés szempontjából kevésbé jó megoldás.

Volk és mtsai (2016) esetében a módszer lényegében ugyanez, Batinić és Schmidt (2018) azonban továbblép: nemcsak hogy odailleszti az elvált igekötő lemmáját a hozzá tartozó igéhez, hanem *törli* az igekötő lemmáját az eredeti helyéről. Emögött az a megfontolás húzódhat meg, hogy a korpusz semmilyen részletét ne duplikáljuk az eredetihez képest. Amint látni fogjuk, jelen tanulmányban még tovább lépünk a keresést megkönnyítő explicit annotáció felé.

Elmondhatjuk, hogy még a szintaktikai elemzőkre épülő keresőkben (Zsibrita és mtsai, 2017) sem valósul meg feltétlenül a könnyű kereshetőség. Pusztán attól, hogy az ige-igekötő él be van húzva, nem lehet minden további nélkül egyben kezelni az elvált és a tapadó alakokat.

3. Módszer

3.1. Igekötő-kapcsolás

Az igekötő-kapcsolás megvalósításához el lehet indulni nyers, morfológiailag elemzett vagy akár szintaktikailag elemzett korpuszból. Egy olyan gazdag morfológiájú nyelvben, mint a magyar, az elsőt nem látjuk célravezetőnek. Az igekötő-kapcsolásban nagyon gyakoriak az egyszerű esetek (pl. '*vesszenek össze*'), viszont számos fajta nemtriviális eset is előfordul (pl. '*el lehetne nem-költői módon is mondani*'). Szintaktikailag elemzett korpuszból kiindulva a feladat triviális lenne – egyetlen dependenciakapcsolatot kell kifejteni –, ehhez azonban fel kell tennünk a szintaktikai elemzés hibátlanságát.

Azt gondoljuk, hogy ezt általában nem tehetjük meg, mert a szintaktikai elemzők főként az egyszerű eseteket kezelik jól. Ha például nagyon távolra kerül egymástól az ige és az igekötő, gyakrabban tévesztenek. Úgy tűnik – és ezt a 4.1. részben olvasható méréseink is alátámasztják –, az igekötő-kapcsolás nem olyan természetű feladat, amiben segít a szintaktikai elemzés. Emiatt, valamint mások hasonló (Sass, 2009; Recski, 2011; Batinić és Schmidt, 2018) megközelítéseire is építve, illetve hogy egy viszonylag kis erőforrásigényű megoldást találjunk, a morfológiailag elemzett korpuszból való kiindulás mellett döntöttünk.

A feldolgozás „irányát" tekintve két lehetőség van. Kívülről befelé esetben a (tag)mondatból kiindulva szűkítjük a szóba jövő tokenek körét, majd mikor már csak egy ige és egy igekötő marad, akkor összekapcsoljuk őket. Belülről kifelé esetben pedig az ige és az igekötő közötti elemeket vizsgáljuk, és ha megfelelő kombinációt találunk, akkor kapcsolunk. Négy ok miatt választottuk az utóbbit: (1) a segédigés esetek (pl. '*haza akarok menni*') kezelése könnyebb; (2) nem szükséges külön tagmondatszegmentálás; (3) az '*el kell, hogy menjek*' típusú példák esetében – legalábbis látszólag – tagmondathatáron átívelő igekötő-ige kapcsolat van; (4) könnyebben kezelhető az árva igekötő, akár ellipszis ('*Megnézted? – Meg.*'), akár elemzési hiba ('*meg[/Prev] várnak[/V] se vár*') miatt fordul elő.

A Volk és mtsai (2016)-hoz hasonló olyan megoldások, amelyek csak adott szótárban előforduló igekötő-ige kombinációkat fogadnak el, a magyarban nem alkalmazhatók. Ennek az az oka, hogy produktív szóalkotással tetszőleges számú igekötős igét hozhatunk létre (pl. *megpirospaprikázzák, lekezitcsókolomoztak*). Ezzel kapcsolatban ld. többek között: Ladányi (2007, 2012); Kalivoda (2021a,b).

A megvalósítás során az `e-magyar` rendszer (Indig és mtsai, 2020) tokenizáló, morfológiai elemző és szófaji címkéző (`tok,morph,pos`) moduljának eredményéből indultunk ki. A módszerünkben egyszerűen azokat a lexikai elemeket tekintjük igekötőnek, amelyek az automatikus morfológiai elemzés során `[/Prev]` címkét kapnak, ezt a címkét nem bíráljuk felül, így a ragos névmásként is értelmezhető igekötők (pl. '*rá*', '*hozzá*') hibás elemzés esetén kimaradnak. (Az igekötő-állomány meghatározásának problémáiról ld. többek között Jakab (1976); Komlósy (1992); Forgács (2005); Kerekes (2011); Kalivoda (2021b) munkáit.) A feladat megoldásakor a gépi tanulás alkalmazását az ehhez szükséges mennyiségű gold standard annotált tanítóadat hiánya eleve kizárta, de egyébként is alkalmasnak látjuk a szabályalapú megközelítést esetünkben, így emellett döntöttünk.

A módszert `xtsv`-modulként implementáltuk. Ennek köszönhetően integráns része lehet a sokak által használt `e-magyar` rendszernek, így szélesebb körben is ismertté és alkalmazhatóvá válhat. Jelenleg a modul `xtsv`-modulként önállóan működik, és tervezzük az `e-magyar` rendszerbe való közvetlen integrációját.

Az algoritmus kialakításának kiindulópontját az MNSZ2-ben (Oravecz és mtsai, 2014) végzett igekötős keresések alkották. A munka első szakaszában iteratívan jártunk el: a minél jobb fedés érdekében minden lépésben a még nem kezelt esetek körében leggyakoribbnak látszó szórendi mintázatot próbáltuk megragadni egy CQL-kifejezéssel. Ha a kapott találatok tisztasága megfelelő volt, az adott keresőkifejezést igekötő-kapcsolási szabállyá fogalmaztuk át; ellenkező esetben finomítottuk a keresőkifejezést további, az `msd` (morfológiai elemzés) és a `word` (szóalak) attribútumokra vonatkozó feltételek hozzáadásával. Egyszerű példával illusztrálva: egy tipikus igekötőre (pl. '*szét*') indított keresés találatain jól látszik, hogy ennek a leggyakoribb szórendi mintázata az ige+igekötő, pl. '*szedte szét*'. Erre a mintázatra rákeresve (`[msd="IGE.*"] [word="szét"]`) azonban azt látjuk, hogy a találatokban sok olyan van, jelesül sok '*volna szét*', amelyekben az igekötőt nem helyes a tőle közvetlenül balra álló igéhez kapcsolni. Ha a keresési feltételünket ennek megfelelően finomítjuk (`[msd="IGE.*" & word!="volna"] [word="szét"]`), akkor a kapott találatok már eléggé tiszták, így megfogalmazhatjuk azt a kapcsolási szabályt, hogy az igekötővel mindig kapcsoljuk össze a közvetlenül tőle balra álló igét, kivéve, ha az a '*volna*' token. Amint a megfelelő keresések találati számai mutatják, ez a szabály lefedi adott esetben a '*szét*' igekötő MNSZ2-beli összes előfordulásának 77%-át úgy, hogy nagyon ritkán eredményez hibás kapcsolást. Ezzel az iteratív eljárással eljutottunk egy általános korpuszpéldákon mérve már „viszonylag jó" pontossággal és fedéssel működő kiinduló algoritmushoz, amelyet az 1. ábra első három pontjában foglaltunk össze.

A második szakaszban új stratégiával javítottuk az algoritmust: mivel az igekötős szórendi mintázatok túlnyomó részét már kezeltük, összeállítottunk egy olyan adathalmazt (ld. 4.1. rész), amely nagy arányban tartalmazott olyan *„nehéz"* eseteket, amelyeket még nem fedtünk le. Ezt az adathalmazt fejlesztő-validáló adatként használva, az algoritmusunk által itt vétett kapcsolási hibákat (hiányzó és téves kapcsolásokat) elemezve, valamint Kalivoda (2021b) átfogó szórendi elemzéseire támaszkodva bővítettük az algoritmust új, nagyobb fedést biztosító szabályokkal úgy, hogy ezek a szükséges mértékben meg legyenek szorítva, ezáltal a kapcsolás pontosságát ne ronthassák. Ezt az eljárást addig folytattuk, amíg a fejlesztő-validáló adathalmazunkban már csak olyan hibásan kapcsolt esetek maradtak, amelyek kezeléséhez már nem célravezető pusztán morfológiai címkéket használni. A végső algoritmus az 1. ábrán látható. Elsősorban rossz központozású szövegekben fordul elő, hogy a szabályok alapján egy igekötő egy tőle balra és egy tőle jobbra lévő igéhez is kapcsolható lenne. Ilyenkor a következő szabállyal döntünk: ha balra és jobbra ugyanolyan távolságra van a két ige, akkor a jobbra lévő igéé lesz az igekötő, különben a közelebbié.

A létrejött `emPreverb` modul Linux környezetben futtatható Python-kódja elérhető a https://github.com/ril-lexknowrep/emPreverb címen, ahol az algoritmus működése részleteiben tanulmányozható.

Az igei természetű szavakon (igék, igenevek és deverbális főnevek) végighaladva, háromszavas ablakban vizsgálódva a következőket hajtjuk végre.
Az előkészítő lépésben kezeljük az eleve igekötős alakokat:
0. '*rá**ordít***'

Ezután az alábbi szabályoknak megfelelő esetekben kapcsoljuk az igekötőt.
A kiinduló algoritmus szabályai:
1. '*meg kell **próbálni***'
2. '***szivárogtatta** ki*'
3. '***tudtam** csak meg*'

A végső algorimus további szabályai:
4. '*meg van **győződve***'
5. '*be nem **tartásának***'
6. '*meg lehetett volna **küldeni***'
7. '*miért **szólna** ő ebbe bele*'
8. '*fel kell, hogy **vállalja***'
9. '*be **kászálódott** az ülésre*' (rossz helyesírás kezelése)

Egyéb esetben nincs kapcsolás, nem igekötős az ige.

1. ábra: Az igekötő-kapcsolás algoritmusa. Az igekötők és az **igék** jelölése. A kiinduló algoritmus a 0–3. lépéseket foglalja magában, a végső az összes szabályt. Az előkészítő lépésben csupán az történik, hogy betoldunk a morfológiai annotáció elejére egy `[/Prev]` kódot. Erre azért van szükség, mert az `e-magyar` nem jelöli a morfológiai kódban az igekötősséget.

3.2. Reprezentáció a korpuszban

Az igekötő-kapcsolás korpuszbeli annotációjának kidolgozásakor a *kényelmes kereshetőség*et tartottuk szem előtt elsődleges szempontként. Ezt az segíti legjobban, ha minél explicitebben fogalmazzuk meg az egyes információkat az annotációban, akár redundáns módon. A megoldásunk hasonlít Batinić és Schmidt (2018)-éhoz, amennyiben az igekötő lemmáját az alapige lemmája elé írjuk, és az igekötő tokenjéből töröljük az eredeti lemmát. Emellett azonban az általunk javasolt annotáció részletesebb, informatívabb. Az '*elő se jön*' példa esetében az algoritmus (1. ábra, 1. szabály) megállapítja, hogy kapcsolnunk kell az igekötőt az igéhez. Nézzük, hogyan jelenik ez meg a kidolgozott annotációs sémában.

1. Az igénél a '*jön*' lemma helyett '*előjön*'-t szerepeltetünk: `lemma=előjön`.
2. A morfológiai kód elejére egy `[/Prev]` kód kerül, ami az igekötősséget jelöli: `xpostag=[/V]...` → `xpostag=[/Prev][/V]...`
3. Az szóösszetételt megjelenítő attribútumba kerül a lemma az igekötőhatár megjelölésével: `compound=elő#jön`
4. A `prev` nevű új mezőben az igénél elkönyveljük, hogy az igekötő eredetileg tapadó (`pfx`) volt vagy elvált (`sep`), illetve az igekötőnél azt, hogy sikerült-e igéhez kapcsolni (`conn`). Példánkban az '*elő*' `prev=conn`, a '*jön*' `prev=sep` lesz.
5. A kapcsolt igekötőnek töröljük az eredeti lemmáját (`lemma=""`).
6. Végül az erre szolgáló `previd` mezőben az igét és az igekötőt is ellátjuk egy közös indexszel: `previd=`n, ahol n egy folyamatosan növekvő egész szám; a

`prevpos` mezőben pedig feltüntetjük az igekötő igétől tokenben számított pozícióját előjeles formában.

Az annotációt az 1. táblázatban foglaljuk össze. Megjegyezzük, hogy a fejlesztés alatt álló `emCompound` modulról, melynek a feladata az, hogy az összetételi határokat megjelölő annotációt adjon a szavakhoz, jelen tanulmányban nem számolunk be részletesen, ugyanakkor egy verzióját közzétesszük. Most nekünk csak azért fontos, mert ez a modul adja meg az igekötő-ige határt a tapadó esetben, ez után következik az `emPreverb` modul futtatása, ami az összes többi feladatot végzi, beleértve az igekötő-ige határ megjelölését az elvált esetben.

form	lemma	xpostag	compound	prev	previd	prevpos
1.						
eloldozódott	eloldozódik	[/V][...]				
→						
eloldozódott	eloldozódik	[/Prev][/V][...]	el#oldozódik	pfx		
2.						
tér	tér	[/V][...]				
vissza	vissza	[/Prev]				
→						
tér	visszatér	[/Prev][/V][...]	vissza#tér	sep	7	+1
vissza	∅	[/Prev]	∅	conn	7	

1. táblázat. Az igekötő-kapcsolás korpuszannotációjának bemutatása. Az eredeti korpuszpélda, majd → után az igekötő-kapcsolás utáni állapot látható. Az igekötő-kapcsolás során az ismertetett `compound`, `prev`, `previd` és `prevpos` mezőkkel bővül az annotáció. Az első példa a tapadó igekötő esete, a második pedig az elvált.

4. Ki#értékelés

4.1. Az igekötő-kapcsolás teljesítménye

Az igekötő-kapcsoló algoritmus kiértékelését különböző tényezők nehezítették:

1. Nem állt rendelkezésünkre előzetesen olyan annotált és *hibátlannak feltételezhető* adathalmaz, amelyet *kifejezetten az igekötő-kapcsolás* problémáira tekintettel állítottak volna össze. Ez azért okoz gondot, mert – amint már utaltunk rá – az igekötő-előfordulások túlnyomó többségének az illesztése majdnem triviális, a legegyszerűbb algoritmusokkal is viszonylag nagy pontossággal megoldható. Az ilyen megoldásokkal nem kezelhető „nehezebb” és egyszersmind ritkább szintaktikai mintázatokban szereplő igekötő-ige párok kapcsolásának sikeressége nehezen mérhető az általános korpuszokon, mivel a nagyszámú „könnyű”, így szempontunkból irreleváns eset elnyomja a kisszámú releváns esetet a mérési eredményekben.
2. Mivel az `e-magyar` rendszer által generált morfológiai elemzések szolgáltak az igekötő-kapcsoló algoritmus bemenetéül, foglalkozni kellett azzal a problémával, hogy ezek az elemzések hibásak lehetnek, így például a rendszer igekötőket tévesen határozószóként (pl. ‘*oda*’) vagy ragozott főnévként (‘*végig*’) elemzett az adott szövegkörnyezetben.

3. Mivel nem egyes tokenek, hanem tokenpárok automatikus annotálását kellett kiértékelni, nem teljesen kézenfekvő, hogy mi számítson helyes, illetve helytelen címkézésnek, valamint az utóbbiakat miként értékeljük ki.

Ezeket a problémákat az alábbiak szerint oldottuk meg.

Tesztadatok Fontosnak tartottuk, hogy olyan statisztikailag minél reprezentatívabb korpuszt használjunk mind az igekötő-kapcsoló algoritmus fejlesztésére, mind teljesítményének mérésére, amely a megoldandó feladattal kapcsolatban kezelendő szintaktikai mintázatokat a tényleges előfordulásaik arányában tartalmazza. Ezt szem előtt tartva két kis tesztkorpuszt állítottunk össze az MNSZ2-ből véletlen mintavétellel. A minta elemeiként egész mondatokat kértünk le. Ugyan az alábbiakban a tömörség kedvéért mindig igékről, illetve ige-igekötő párokról beszélünk, valójában a tesztkorpuszainkban nemcsak igéket, hanem más igei természetű szavakat (igeneveket és deverbális főneveket) is annotáltunk az igekötőhöz tartozó „ige"-ként, ha az igekötő éppen ilyentől vált el.

1. Egy kb. 500 mondatot tartalmazó *„általános"* véletlen minta, amelyet az `[msd="IK"]` keresés eredményeként kaptunk, majd úgy szűrtünk, hogy mondatonként pontosan egy ige-igekötő pár szerepeljen benne. Terjedelme mintegy 13000 token. Az ige-igekötő párokat a példák ellenőrzését követően kézzel, koindexálással megjelöltük összetartozóként a nyers szövegben.
2. Egy kb. 600 mondatot tartalmazó, mintegy 19000 token terjedelmű *„nehéz"* véletlen minta, amelyet az `[msd!=".*IGE.*"] [msd="IK"]` kereséssel kaptunk. Ez tehát olyan mondatokból áll, amelyek tartalmaznak legalább egy olyan igekötő-előfordulást, amelytől közvetlenül balra nem ige áll. A több igekötő-ige párt tartalmazó mondatokat megtartottuk, és a bennük található összes összetartozó igekötő-ige párt kézzel annotáltuk a nyers szövegben, kivéve a triviálisnak tekintett igekötő-ige párokat, ahol az igekötőtől közvetlenül balra áll az igéje (pl. '*nézett ki*'). Ha a találatként kapott mondatokban szerepelt olyan (kevésbé triviálisan kezelhető) segédigés szerkezet is, amelyet a keresési feltétel alapján nem kaptunk volna találatként (pl. '*nézett volna ki*'), akkor ezekben is összetartozóként jelöltük meg az igekötőt és az alapigét. Az elliptikus szerkezetben szereplő, igéhez nem kapcsolható igekötőket külön jelöléssel láttuk el. Mellérendelő szerkezetekben megengedtük, hogy tapadó igekötős igéhez is tartozhasson másik, elvált igekötő (pl. '*át- meg átjárta*'); illetve hogy egy igéhez több elvált igekötő tartozzon (pl. '*mutat vissza a város középkori történetére, vissza a ködszerű múltra*') Az utóbbi esetben az azonos igéhez tartozó igekötők azonos indexet kaptak, ugyanakkor feltételezzük, hogy egy igekötő mellérendelő szerkezetben sem tartozhat több mint egy igéhez.

Mindkét korpusz esetében nagyobb mintából indultunk ki eredetileg, és ebből eltávolítottuk (1) a különbözőképpen hibás vagy szempontunkból irreleváns (pl. ékezetek nélküli, hibásan beszúrt szóközöket tartalmazó, archaikus nyelvezetű vagy írásmódú stb.) mondatokat, amelyek esetében az `e-magyar` morfológiai elemzése vagy szófaji címkézése várhatóan kudarcot vallott volna. (2) Emellett

szintén töröltük az igekötőt valójában nem tartalmazó, azaz az MNSZ2-ben tévesen igekötősként szereplő mondatokat, ahol nem volt mit kapcsolni. (3) Töröltük továbbá a koindexálást azokról az igekötő-ige párokról, amelyek esetében az `e-magyar` szófaji elemzője tévesen nem igekötőként címkézte fel a tényleges igekötőt. Mindezeket a lépéseket az indokolja, hogy a kiértékelés során *célzottan az igekötő-kapcsoló algoritmusok helyességét* akartuk megítélni önmagában, és ennek érdekében irrelevánsként ki akartuk zárni az olyan kapcsolási hibák lehetőségét, amelyek az `e-magyar` elemzésének hibájából fakadnak. Nem ellenőriztük ugyanakkor, hogy az `e-magyar` helyes morfológiai és szófaji címkéket rendel-e a mondat egyéb, különösen az igekötő és az ige közé beékelődött szavaihoz, ugyanis úgy véljük, hogy az igekötő-kapcsoló algoritmusnak lehetőleg elég robusztusnak kell lennie ahhoz, hogy az ilyen elemzési hibák jelenlétében is minél pontosabban működjön.

A kétféle tesztadathalmaz együttes használatát az indokolja, hogy *egyfelől* olyan megoldást keresünk, amely a kapcsolás tekintetében nehéz szintaktikai mintázatokat is helyesen kezeli – ezek csak a *„nehéz”* halmazon mérhetőek hatékonyan –, *másfelől* az *„általános”* halmazon történő kiértékelés révén biztosítjuk, hogy az eljárás az egyszerű esetekben gazdag, sima magyar szövegen is működjön.

A *„nehéz”* eseteket tartalmazó adathalmazt két diszjukt részre osztottuk. Az egyik részt ($N = 1292$ igekötővel) az igekötő-kapcsoló algoritmus fejlesztése, a szabályok finomhangolása, hibaelemzése során fejlesztő-validáló halmazként használtuk, míg a másik részt ($N = 376$ igekötővel) teszthalmazként félretettük, és csak az alább ismertetett mérésekre használtuk. Az *„általános”* adathalmazt kizárólag teszthalmazként használtuk, a fejlesztés során semmilyen módon nem használtuk fel.

A teljesítménymérés elvei Mind a saját algoritmusunk, mind a vele összehasonlításképpen vizsgált egyéb eljárások kiértékelése során az alábbi definíciókat alkalmaztuk. Nem ige-igekötő párokat számoltunk, hanem *igekötőket.* Pontosan azokat a tokeneket tekintettük igekötőnek, amelyeket a tesztadatainkban a fentebb összefoglalt elveket követve kézzel igekötőként jelöltünk meg. Egy igekötőt *pozitívnak* tekintünk, ha az algoritmus kapcsolt hozzá igét. Ezen belül *helyes pozitívnak* tekintettük, amennyiben pontosan azt az igei természetű tokent kapcsolta hozzá, amellyel koindexáltuk a kézi annotáció során, és *hamis pozitívnak,* ha bármely más tokent kapcsolt hozzá. Megfordítva egy igekötőt *negatívnak* tekintünk akkor, ha az algoritmus *nem* kapcsolt hozzá más tokent. Ezen belül *helyes negatívnak,* ha valóban nem kellett hozzá kapcsolni semmit (ez elliptikus szerkezetekben áll fenn), és *hamis negatívnak,* ha a kézi annotáció szerint kellett volna kapcsolni hozzá tokent. Az említett négy mérőszám alapján számítottuk a szokásos pontosság, fedés és F_1 teljesítménymutatókat. Beszámolunk továbbá az accuracy értékről is, ugyanis ezt hasonlóan relevánsnak gondoljuk az adott feladat összefüggésében.

A saját algoritmusunkat összehasonlítottuk két egyszerűbb igekötő-kapcsoló algoritmussal mint baseline-nal:

- `nearest-verb`: Az igekötőket kapcsoljuk mindig a legközelebbi igéhez, *kivéve* a közvetlenül tőlük jobbra álló igéhez.
- `max2`: Minden igére vizsgáljuk meg, hogy van-e tőlük balra pontosan két token távolságban (tehát egy közbeeső tokennel), illetve jobbra pontosan egy vagy két token távolságban igekötő, és ha igen, akkor kapcsoljuk hozzá.

A `nearest-verb` algoritmus hasonlít Recski (2011, 3.1.) baseline-jához, amely egyszerűen a legközelebbi igéhez kapcsolja az igekötőt minden korlátozás nélkül. Úgy ítéltük meg, hogy ez utóbbi mesterségesen rossz baseline, ugyanis az igekötőtől közvetlenül jobbra álló igéhez a helyesírási normát viszonylag követő korpuszokban (mint amilyen az MNSZ2 túlnyomó része) *szinte soha* nem helyes az igekötőt kapcsolni. Ott ugyanis általában csak a szerző által további lépésekben kizárt segédige jellegű igék állhatnak, amelyek pedig soha nem az igekötő alapigéi (pl. '*ki tudja nyitni*'; '*be lenne zárva*'). Legegyszerűbbnek tehát azt láttuk, hogy a `nearest-verb` baseline-t a fenti nagyon egyszerű korlátozó feltétellel alkalmazzuk.

A `max2` baseline alapötlete, hogy az igétől közvetlenül balra – az imént kifejtett okból – szinte soha nem áll az igekötője, és – amint Kalivoda (2021b, 14., 15., 18., 20. és 21. táblázat) összegzi – az igekötő az összes eset elenyésző részében távolodik el az igéjétől több mint két token távolságra.

A két baseline mellett algoritmusunkat összehasonlítottuk az `e-magyar` rendszer `emStanza` moduljának outputjával is, amely a szintaktikai elemzés részeként (tehát ellentétben a mi módszerünkkel és a két baseline-nal nem pusztán morfológiai információkra támaszkodva) kapcsolja az igekötőket az igéjükhöz. Ennek kapcsán értelemszerűen arra a kérdésre kerestük a választ, hogy a szintaktikai elemzéssel megbízhatóan jobb igekötő-kapcsolást kapunk-e, mint anélkül. A Stanza (Qi és mtsai, 2020) szintaktikai elemző outputjában az igekötő-kapcsolás a `compound:preverb` függőségi reláció formájában közvetlenül megjelenik, így ez az információ további feldolgozás nélkül rendelkezésre áll, és pontossága kézenfekvően kiértékelhető a fenti elveket követve.

a)

annotáció \ auto címke	pozitív	negatív	
helyes	489	0	489
hamis	1	10	11
	490	10	500

b)

annotáció \ auto címke	pozitív	negatív	
helyes	346	6	352
hamis	3	21	24
	349	27	376

2. táblázat. Az a) *„általános”*, illetve a b) *„nehéz”* esetek konfúziós mátrixa.

A mérés eredményei A két tesztadathalmaz mérete $N_{általános} = 500$, illetve $N_{nehéz} = 376$. Algoritmusunk értékeléséhez érdekes lehet egy pillantást vetni az *„általános”* és *„nehéz”* esetek konfúziós mátrixára (2. táblázat). A kiértékelés eredményei a 3. táblázatban láthatók.

Összefoglaló észrevételek Amint az algoritmusunk (vö. 1. ábra) ismertetése kapcsán hangsúlyoztuk, kiemelt célkitűzésünk volt a minél magasabb pontosság biztosítása. A pontosság előtérbe helyezése az eredményekben jól tükröződik, az `emPreverb` ebben a tekintetben mindkét tesztadathalmazon látványosan jobb

	pontosság (%)	fedés (%)	F_1	accuracy (%)
`emPreverb`	**99,80**	98,00	**0,9889**	**97,80**
`nearest-verb`	96,56	**98,76**	0,9765	95,40
`max2`	97,94	96,94	0,9744	95,00
`emStanza`	96,04	90,66	0,9328	87,40

(a) Eredmények az *„általános”* mintán.

pontosság (%)	fedés (%)	F_1	accuracy (%)
99,14	**94,28**	**0,9665**	**93,62**
76,62	93,79	0,8434	73,14
92,28	79,48	0,8540	75,00
90,83	89,24	0,9003	81,91

(b) Eredmények a *„nehéz”* mintán.

3. táblázat. A vizsgált algoritmusok eredményei.

eredményt ér el a három másik megoldáshoz képest. A fedés nem közelíti ezt meg, de összességében nem rosszabb az alternatívákhoz képest. Vegyük észre, hogy a `nearest-verb` baseline fedése triviális módon lesz magas, mivel ez a módszer minden igekötőhöz igyekszik kapcsolni valamit, bármilyen messze is legyen tőle. Ez persze a pontosság rovására megy, ami összességében alacsonyabb F_1 pontszámot eredményez. Ugyan az *„általános”* számok ezt – a fentebb részletezett okok miatt törvényszerűen – kevésbé adják ki, a *„nehéz”* adatok világosan mutatják, hogy az `emPreverb` meggyőzően jobban teljesít az összes többi megoldáshoz képest.

Figyelemreméltóak a Stanza eredményei: ugyan a *„nehéz”* adatokon messze a két baseline felett teljesít (bár hasonlóan messze elmaradva az `emPreverb` számaitól), az *„általános”* adatokon még a „buta” baseline-okhoz képest is minden szempontból nagyon rosszak az eredményei, így praktikusan nem használható az igekötő-kapcsolás automatikus annotálására korpuszokban.

A https://github.com/ril-lexknowrep/hungarian-preverb-corpus oldalon megtalálhatók a kézzel annotált fejlesztő és tesztkorpuszok az annotálási útmutatóval együtt, valamint a teljes kiértékelési környezet, melynek segítségével a fenti kiértékelés reprodukálható.

4.2. A reprezentáció hasznossága

Ebben a fejezetben az igekötő-kapcsolás jelen tanulmány keretében kidolgozott korpuszbeli reprezentációjának, annotációjának hasznosságát mutatjuk be példákon keresztül a 3.2. rész pontjai szerint haladva. Javaslatunk a harmadik, negyedik és hatodik pont tekintetében tud többet a Batinić és Schmidt (2018, 1. táblázat) által bemutatott modellhez képest.

1. Talán a legfontosabb eredmény, hogy az igekötős igék *összes* korpuszbeli találatát megkaphatjuk – tapadástól/elválástól függetlenül – egy egyszerű, pusztán a lemmára irányuló CQL lekérdezéssel: `[lemma="elõjön"]`, és ennek köszönhetően gyakorisági listát is egyszerűen készíthetünk a találatokból.
2. Azáltal, hogy bővítjük az igekötős igék morfológiai kódját a `[/Prev]` elemmel, könnyen hozzájuthatunk az összes igekötőtlen igéhez: `[xpostag="\[/V\].*"]` akár összes igekötős igéhez: `[xpostag="\[/Prev\]\[/V\].*"]` ismét csak függetlenül attól, hogy az igekötő elválik-e. Utóbbi kitételt minden további megállapításunkhoz is értsük hozzá.
3. A `[compound="meg#.*" & xpostag="\[/Prev\].+"]` lekérdezéssel megkapjuk az összes ‘*meg*’ igekötős igét. Megjegyezzük, hogy ennek megvalósulásához a tapadó esetben az `emCompound` modulra, az elvált esetben pedig az `emPreverb` modulra van szükség.
4. A `prev` mezőre támaszkodva lekérdezhetjük adott igekötős ige összes tapadó: `[lemma="elõjön" & prev="pfx"]` vagy összes elvált: `[lemma="elõjön" &`

`prev="sep"]` alakját, vizsgálhatjuk az igék elválási hajlandóságát. Általánosan, azaz minden igére egyben, a `[xpostag="\[/Prev\]\[/V\].*"]` lekérdezéssel, majd a belőle `lemma+prev` szerint készített gyakorisági lista révén kaphatjuk ezt meg.

5. Az odakapcsolt igekötőket – az üres lemmának köszönhetően – a `[lemma="" & xpostag="\[/Prev\]"]` lekérdezéssel, az árván maradt igekötőket pedig a `[lemma=".+" & xpostag="\[/Prev\]"]` lekérdezéssel gyűjthetjük össze. Az árván maradt igekötők átalában ellipszis vagy hibás morfológiai annotáció – például egy igekötőként azonosított határozószó – eredményei, ahogy erről már volt szó.
6. A `prevpos` mezőből készített gyakorisági lista által megállapíthatjuk az igekötő eltávolodásának eloszlását adott/összes ige és adott/összes igekötő vonatkozásában is.

Ha arra vagyunk kíváncsiak, hogy adott igekötő mennyire szeret elválni, akkor ez a lekérdezés: `[prev="(sep|pfx)" & compound="meg#.*"]` és a `compound+prev` szerint belőle készített gyakorisági lista lesz segítségünkre. Ha pedig úgy tesszük fel a kérdést, hogy a magyar nyelvben általában mennyire szeretnek elválni az igekötők, akkor a lekérdezést a következőre cserélve: `[prev="(sep|pfx)"]` és `prev` szerinti gyakorisági listát készítve azt kapjuk eredményül, hogy 60-65% a tapadó és 35-40% az elvált igekötő – vö. (Kalivoda, 2021b) 157. oldalán szereplő 41,2%-os adattal.

Ahogy korábban utaltunk rá, a könnyű lekérdezhetőséghez az explicit annotáció visz közelebb. Ezt támasztják alá a fentiek. Az annotáció hasznosságához hozzátartozik, hogy az elmondottak mind egy az egyben implementálhatók a NoSketchEngine korpuszkezelő alatt, mi is így jártunk el a fenti lekérdezések kipróbálásakor.

5. Össze#foglalás

Tanulmányunkban (1) szabályalapú megoldást adtunk a magyar elváló igekötőnek és az igéjének az automatikus egymáshoz rendelésére, és (2) bemutattunk egy modellt az igekötők és igék viszonyára vonatkozó információknak a korpusz annotációjában való olyan megjelenítésére, mely a korpuszban történő keresést a lehető legnagyobb mértékben megkönnyíti. Válogatott nehéz példák alapján készített automatikus módszerünk teljesítménye ($F_1 = 0{,}9889$) meggyőzően jobb a korábbiakénál, és a kiértékelés során azt is megmutattuk, hogy a feladat megoldásakor nem érdemes automatikus szintaktikai elemzésre támaszkodni, a morfológiai elemzésre építő szabályalapú módszer jobb eredményt ad. Számos példával illusztráltuk, hogy a javasolt reprezentációs séma milyen új lekérdezési lehetőségekre nyit kaputt. Minden eszköz és erőforrás elérhető a https://github.com/rillexknowrep oldalról: az igekötő-kapcsoló modul az emPreverb, az összetételi határokat megjelölő modul az emCompound, az annotált korpuszok és a kiértékelő környezet pedig a hungarian-preverb-corpus repozitóriumban. Eredményeink lényegében minden magyar nyelvű lexikográfiai projektben és korpuszalapú nyelvészeti kutatásban hasznosulhatnak a jövőben, ehhez hozzájárulhat az is, hogy az implementáció a közismert `e-magyar` rendszerbe illeszkedik.

Hivatkozások

Batinić, D., Schmidt, T.: Reconstruction of Separable Particle Verbs in a Corpus of Spoken German. In: Rehm, G., Declerck, T. (szerk.) Language Technologies for the Challenges of the Digital Age. pp. 3–10. Springer International Publishing, Svájc, Cham (2018)

Csendes, D., Csirik, J., Gyimóthy, T., Kocsor, A.: The Szeged Treebank. In: Matoušek, V., Mautner, P., Pavelka, T. (szerk.) Proceedings of the 8th International Conference on Text, Speech and Dialogue (TSD 2005). pp. 123–131. Springer LNAI, Berlin, Heidelberg (2005)

Dömötör, A., Gugán, K., Novák, A., Varga, M.: Kiútkeresés a morfológiai labirintusból – korpuszépítés ó- és középmagyar kori magánéleti szövegekből. Nyelvtudományi közlemények 113, 85–110 (2017)

Forgács, T.: Grammatikalizálódás az igekötők körében. In: Oszkó, B., Sipos, M. (szerk.) Uráli grammatizáló. pp. 88–116. MTA Nyelvtudományi Intézet, Budapest (2005)

Indig, B., Sass, B., Mittelholcz, I.: The xtsv Framework and the Twelve Virtues of Pipelines. In: Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S. (szerk.) Proceedings of the 12th Language Resources and Evaluation Conference. pp. 7044–7052. European Language Resources Association, Marseille (2020)

Ittzés és mtsai: A magyar nyelv nagyszótára (2006–2021), `https://nagyszotar.nytud.hu`

Jakab, I.: A magyar igekötők állományi vizsgálata. No. 91 in Nyelvtudományi Értekezések, Akadémiai Kiadó, Budapest (1976)

Kalivoda, Á.: Az igekötők produktív kapcsolódási mintái. Argumentum 17, 56–82 (2021a)

Kalivoda, Á.: Igekötős szerkezetek a magyarban. Doktori (PhD) értekezés, Pázmány Péter Katolikus Egyetem, Bölcsészet- és Társadalomtudományi Kar, Nyelvtudományi Doktori Iskola, Budapest (2021b)

Kerekes, J.: Az igekötők meghatározásának problémái. In: Gécseg, Zs. (szerk.) LingDok10: Nyelvész-doktoranduszok dolgozatai. pp. 109–131. Szegedi Tudományegyetem Nyelvtudományi Doktori Iskola, Szeged (2011)

Komlósy, A.: Régensek és vonzatok. In: Kiefer, F. (szerk.) Strukturális magyar nyelvtan 1., Mondattan, pp. 299–527. Akadémiai Kiadó, Budapest (1992)

Köper, M.: Computational approaches for German particle verbs: compositionality, sense discrimination and non-literal language. Doktori PhD értekezés, Institut für Maschinelle Sprachverarbeitung der Universität Stuttgart, Stuttgart (2018)

Ladányi, M.: Produktivitás és analógia a szóképzésben: elvek és esetek. Tinta Könyvkiadó, Budapest (2007)

Ladányi, M.: Igekötős igék kapcsolódási mintázatai. Vizsgálati lehetőségek. In: Tolcsvai Nagy, G., Tátrai, Sz. (szerk.) Konstrukció és jelentés. Tanulmányok a magyar nyelv funkcionális kognitív leírására, pp. 71–84. Eötvös Loránd Tudományegyetem, Budapest (2012)

Lipp, V., Simon, L.: Towards a new monolingual Hungarian explanatory dictionary: overview of the Hungarian explanatory dictionaries. Studia lexicographica 15(29), 83–96 (2021)

Novák, A., Laki, L.J., Novák, B., Dömötör, A., Ligeti-Nagy, N., Kalivoda, Á.: Egy magyar nyelvű kérdezőrendszer. In: XV. Magyar Számítógépes Nyelvészeti Konferencia. pp. 83–95. Szegedi Tudományegyetem, Informatikai Tanszékcsoport, Szegedi Tudományegyetem, Informatikai Tanszékcsoport, Szeged (2019)

Oravecz, Cs., Váradi, T., Sass, B.: The Hungarian Gigaword Corpus. In: Calzolari, N., Choukri, K., Declerck, T., Loftsson, H., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., Piperidis, S. (szerk.) Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC 2014). pp. 1719–1723. European Language Resources Association (ELRA), Reykjavik, Iceland (2014)

Qi, P., Zhang, Y., Zhang, Y., Bolton, J., Manning, C.D.: Stanza: A Python natural language processing toolkit for many human languages. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations (2020)

Recski, G.: A sekély mondattani elemzés további lépései. In: Tanács, A., Vincze, V. (szerk.) VIII. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY 2011). pp. 113–118. Szegedi Tudományegyetem, Informatikai Tanszékcsoport, Szeged (2011)

Sass, B.: „Mazsola" – eszköz a magyar igék bővítményszerkezetének vizsgálatára. In: Váradi, T. (szerk.) Válogatás az I. Alkalmazott Nyelvészeti Doktorandusz Konferencia előadásaiból. pp. 117–129. MTA Nyelvtudományi Intézet, Budapest (2009)

Sass, B.: Igei szerkezetek gyakorisági szótára – egy automatikus lexikai kinyerő eljárás és alkalmazása. Doktori PhD értekezés, Pázmány Péter Katolikus Egyetem, Budapest (2011)

Sass, B., Miháltz, M., Kundráth, P.: Az e-magyar rendszer GATE környezetbe integrált magyar szövegfeldolgozó eszközlánca. In: Vincze, V. (szerk.) XIII. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY 2017). pp. 79–90. Szegedi Tudományegyetem, Informatikai Tanszékcsoport, Szeged (2017)

Simon, E., Sass, B.: Nyelvtechnológia és kulturális örökség, avagy korpuszépítés ómagyar kódexekből. Általános Nyelvészeti Tanulmányok 24, 243–264 (2012)

Vadász, N., Kalivoda, Á., Indig, B.: Ablak által világosan – vonzatkeret-egyértelműsítés az igekötők és az infinitívuszi vonzatok segítségével. In: Vincze, V. (szerk.) XIII. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY 2017). pp. 3–12. Szegedi Tudományegyetem, Informatikai Intézet, Szeged (2017)

Vadász, N., Kalivoda, Á., Indig, B.: Egy egységesített magyar igei vonzatkerettár építése és felhasználása. In: Vincze, V. (szerk.) XIV. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY 2018). pp. 3–15. Szegedi Tudományegyetem, Informatikai Intézet, Szeged (2018)

Volk, M., Clematide, S., Graën, J., Ströbel, P.: Bi-particle Adverbs, PoS-Tagging and the Recognition of German Separable Prefix Verbs. In: Dipper, S., Ne-

ubarth, F., Zinsmeister, H. (szerk.) Proceedings of the 13th Conference on Natural Language Processing (KONVENS 2016). Bochumer Linguistische Arbeitsberichte, vol. 16, pp. 3–10 (2016)

Zsibrita, J., Farkas, R., Vincze, V.: Függőségi elemzésen alapuló magyar nyelvű keresőrendszer. In: Vincze, V. (szerk.) XIII. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY 2017). pp. 363–369. Szegedi Tudományegyetem, Informatikai Tanszékcsoport, Szeged (2017)