A HuSpaCy és e-magyar elemzőláncok teljesítményének átfogó összehasonlítása országgyűlési szövegeken: a tokenizálástól a függőségi elemzésig

Skrabák Boglárka[1], Ligeti-Nagy Noémi[2]

[1]ELTE Informatikai Kar
[2]HUN-REN Nyelvtudományi Kutatóközpont
skrabakbogi@gmail.com
ligeti-nagy.noemi@nytud.hun-ren.hu

Kivonat A magyar nyelv számítógépes feldolgozásához megbízható elemzőrendszerekre van szükség, amelyek különböző nyelvi szinteken képesek pontos kimenetet biztosítani. Jelen tanulmány célja, hogy részletes összehasonlítást nyújtson két kiemelkedő magyar nyelvi elemzőrendszer, a HuSpaCy és az e-magyar teljesítményéről több szempontból, ideértve a tokenizálást, a szófaji címkézést, a morfoszintaktikai elemzést és a névelem-felismerést. Az elemzéseinket a magyar országgyűlési jegyzőkönyvek egy szövegrészén végeztük, amely lehetőséget nyújtott a rendszerek tesztelésére formailag és nyelvileg változatos szövegeken. Eredményeink segítséget nyújthatnak a nyelvi feldolgozó eszközök felhasználóinak abban, hogy a specifikus nyelvi alkalmazási igényeiknek legmegfelelőbb rendszert válasszák. A tanulmány rámutat az elemzők erősségeire és hiányosságaira, melyek alapján továbbfejleszthetők a magyar nyelvre irányuló nyelvtechnológiai megoldások.
Kulcsszavak: nyelvfeldolgozás, magyar nyelvi elemző, HuSpaCy, e-magyar, tokenizálás, morfoszintaktikai elemzés, névelem-felismerés

1. Bevezetés

A nyelvi elemzőeszközök terén a HuSpaCy (Orosz és mtsai, 2022, 2023) és az e-magyar (Váradi és mtsai, 2017; Indig és mtsai, 2019; Simon és mtsai, 2020) rendszerek kiemelkednek, mint a magyar nyelv feldolgozására alkalmas nyílt forráskódú megoldások. A HuSpaCy egy korszerű, könnyen használható, gyors elemzőrendszer, amely a SpaCy keretrendszerre épülve (Honnibal és mtsai, 2020) képes a magyar nyelv morfoszintaktikai feldolgozására. Az e-magyar rendszer egy komplex, többkomponensű elemzői *pipeline*, amely a magyar nyelv specifikus sajátosságait is figyelembe veszi.

A magyar nyelvre optimalizált ipari nyelvfeldolgozó eszközök száma korlátozott, és csak néhány átfogó rendszer érhető el a fentieken kívül. Ezek közé tartozik a magyarlanc (Zsibrita és mtsai, 2013), amely egy Java-alapú, ipari alkalmazásokhoz tervezett eszköz. Bár számos fontos nyelvfeldolgozási funkciót nyújt –

mint a tokenizálás, mondathatárok felismerése, szófaji címkézés, lemmatizálás és függőségi elemzés –, hiányossága a névelem-felismerés és a szóbeágyazások támogatásának hiánya.

A magyar nyelvspecifikus rendszerek mellett nemzetközi eszközök, mint a Stanza (Qi és mtsai, 2020) és az UDPipe (Straka, 2018), szintén relevánsak a nyelvfeldolgozás területén. Ezek több nyelvet támogatnak, és képesek nyers szövegek morfológiai és szintaktikai elemzésére. Ugyanakkor ezek az eszközök sem nyújtanak névelem-felismerési lehetőséget, és teljesítményüket jelentősen korlátozza a nyilvánosan elérhető annotált korpuszok kis mérete.

Jelen tanulmány célja, hogy összehasonlítsa a fent említett két elemzőrendszert, a HuSpaCy-t és az e-magyart, különböző feldolgozási szinteken, beleértve a tokenizálást, a szófaji címkézést, a morfoszintaktikai elemzést és az egyéb nyelvi feldolgozási szinteket. Vizsgálatunk fő motivációja egy most készülő korpusz,[1] melyet részletes elemzéssel kell ellátnunk. A megfelelő elemző kiválasztásához elengedhetetlen egy alapos, módszeres összehasonlítás. Az összehasonlítás eredményei ezen felül hozzájárulhatnak a magyar nyelvtechnológiai alkalmazások fejlesztéséhez és optimalizálásához, valamint segíthetnek az alkalmazók számára a legmegfelelőbb elemzőrendszer kiválasztásában.

2. Kapcsolódó irodalom

A **HuSpaCy** egy ipari felhasználásra szánt, nagy hatékonyságú eszközlánc, amely a spaCy keretrendszerre építve biztosít alapvető nyelvfeldolgozási funkciókat, mint a tokenizálás, szófaji címkézés, lemmatizálás, dependenciaelemzés és tulajdonnév-felismerés (NER) (Orosz és mtsai, 2022, 2023). Célja a gyors és pontos elemzés erőforrástakarékos környezetben. A szerzők mérései alapján kimagasló pontosságot ér el a tokenizálásban és a mondathatárok felismerésében. Előbbiben szinte minden rendszerrel összehasonlítva a legjobb eredményt mutatta (99,89%-os F1-érték). A mondathatárok felismerése terén is kiemelkedő (97,66%), bár itt az e-magyar valamivel jobbnak bizonyult. A szófaji címkézésben a HuSpaCy különösen jól teljesít, az Universal Dependencies (UD) adatain tanított modell 94,7%-os pontosságot ér el, míg a Szeged Korpusz (Csendes és mtsai, 2004) adataival továbbfejlesztett modell 96,58%-ot. A függőségi elemzésben a HuSpaCy eredményei nem érik el a Stanza által elért legmagasabb pontszámokat. Lemmatizációs modulja az egyik legpontosabb a vizsgált rendszerek között. A Szeged Korpusz teljes adatkészletén tanított modell 95,53%-os pontosságot ért el, ami meghaladja az e-magyar teljesítményét (94,94%) és jelentősen jobb a UDPipe (88,5%) eredményeinél. A névelem-felismerés terén azonban nem éri el a BERT-alapú modellek pontosságát.

Az **e-magyar** rendszer célja a magyar nyelv elemzésére alkalmas eszközök integrálása egy moduláris, kutatásorientált keretbe. A legújabb, emtsv verzió (Indig és mtsai, 2019) teljes modularitást kínál, amely lehetővé teszi, hogy különböző modulok (pl. tokenizálás, morfológiai elemzés) különálló egységekként

[1] Részletek a tanulmány végső változatában.

működjenek, és az elemzőlánc bármely pontján kimenetet biztosítsanak. A szerzők által közölt teszteredmények alapján az emtsv hasonlóan teljesít más magyar nyelvi elemző rendszerekhez. Az elemzési lánc az **emToken** modul által végzett tokenizálással és mondathatárok meghatározásával kezdődik. Ezt követi a morfológiai elemzés az **emMorph** (Novák, 2014; Novák és mtsai, 2016) és az **emLem** modulokkal. Az egyértelműsítést a POS címkéző modul (**emTag**, Orosz és Novák, 2013) végzi. A morfológiai elemzés után a mondattani elemzés két irányban folytatódik: az **emDep** modul függőségi struktúrákat tár fel, míg az **emCons** modul az összetevők hierarchiáját állapítja meg. A szövegek további feldolgozása során az **emChunk** főnévi csoportokat azonosít, míg az **emNer** modul tulajdonneveket és speciális kifejezéseket jelöl. A rendszer tartalmaz kiegészítő modulokat is, mint például az **emZero**, amely zérónévmásokat illeszt be, és az **emTerm**, amely több szóból álló kifejezéseket annotál.

3. Módszer

Az összehasonlító elemzés alapjául a 2022. 06. 20-ai országgyűlési jegyzőkönyv szövege szolgált. A szöveg 2498 sorból, 290 606 karakterből áll, és nyelvezetét, formáját tekintve változatos. Az igényesen megfogalmazott, retorikus élőbeszéd mellett tartalmaz természetes élőbeszédből eredő kifejezéseket, mondatszerkezeteket is, a lejegyzésből adódó helyesírási hibákat, valamint a jegyzőkönyvi formából eredő nem nyelvi elemeket is (például felszólalók nevei, időpontok, sorszámok és paragrafusok jelzése, telefonszám, postacím, szövegtagoló karakterek). Ezek mind nagyszerű lehetőséget nyújtottak arra, hogy nem tipikus, de írott szövegben mégis előforduló nyelvi helyzetekben is teszteljük az elemzők teljesítményét. Választásunkat az is indokolja, hogy kifejezetten a készülő korpuszunk számára keressük a legmegfelelőbb elemzőláncot, így indokoltnak tűnik egy általunk gyűjtött, és később felhasználandó szövegen összevetni őket, nem pedig meglévő, *gold standard*-nek tartott korpuszokon, amelyek jellemzően egyszerre csak egy-egy elemzési szint validálását tennék lehetővé. Így biztosítjuk, hogy olyan szövegen teszteljük a két elemzőt, amit azok korábban biztosan nem láttak.

Érdemes külön kiemelni a *whitespace* karakterek szerepét. A dokumentumban a szóközök mellett a leggyakoribb *whitespace* karakter a sortörés („\n”), ugyanis a bemeneti szöveg bekezdéseinek tagolását általában kettő vagy több egymást követő sortörés valósítja meg. Ezek nagymértékben befolyásolják az elemzési eredményeket a tokenizálás és a dependencia szintjén; kezelésük az elkészült összehasonlító program sarkalatos pontja. Általánosságban elmondható, hogy a szövegben nagy mennyiségben és sokféle formában előforduló központozási, numerikus és *whitespace* karakterek nagyban megnehezítik a megfelelő összehasonlítás elkészítését mind a HuSpaCy és az e-magyar, mind pedig az őket összehasonlító alkalmazás szintjén.

Ez utóbbit a `Launcher_HuSpaCy_emagyar` alkalmazás[2] teszi lehetővé. Ez egy Python nyelven írt programcsomag, amely konzolos alkalmazásként, többféle

[2] `https://github.com/nytud/thesis-works.git`

módban indítható, és a működése során tetszőleges `txt` kiterjesztésű szövegen lefuttatja az elemzőket – vagy csak az egyiket, vagy mindkettőt. Az elemzési eredményeket fájlban eltárolja, valamint opcionálisan a terminálra is kiírja. Mindemellett lekérhető belőle a különböző elemzési szinteken kapott eredmények összehasonlítása egy egyszerű, de az adott elemzési szempont sajátosságaihoz illeszkedő táblázatos formában.

Az alkalmazás indítása a `launcher.py` program segítségével lehetséges. Indításkor parancssori argumentumként meg lehet adni az elemzendő fájl elérési útvonalát, a kívánt elemzőt (`-huspacy` vagy `-emagyar` kapcsoló), illetve az összehasonlítás támogatására szolgáló kapcsolók valamelyikét. Ezek lehetnek arra vonatkozó parancsok, hogy az összesített elemzés megjelenjen-e a terminálablakon is (`-oute` az emagyar, `-outh` a HuSpaCy számára), valamint a nyelvi szint összehasonlítási eredményének megjelenítésére vonatkozó kapcsolók (`-tok`: tokenizálás, `-morph`: morfológia, `-lem`: lemmatizálás, `-pos`: szófaji címkézés, `-dep`: dependencia, `-ner`: névelem-felismerés). Az összehasonlítási kapcsolók értelemszerűen csak akkor lépnek életbe, ha mindkét elemzőt futtatjuk, egyéb esetben nincsen semmilyen hatásuk.

Az alkalmazás (továbbiakban: Launcher) túlmutat az elemzők lefuttatásán: egy egész környezetet valósít meg, amely minél több szempontból támogatja az elemzők nyelvészeti teljesítményének összehasonlítását. Ezek a szempontok a következő kategóriákba sorolhatók: futtatás, logikai egységesítés, különbségek kezelése, teljesítmény javítása, illetve egyszerű teljesítménymetrika.

3.1. Futtatás

A HuSpaCy egy SpaCy-könyvtár formájában működik[3], így kényelmesen használható Python-scripteken belül. Jelen elemzésünkben a hu_core_news_lg HuSPaCy-modellt használjuk. Az e-magyar jelenlegi változata Docker-alkalmazásként működik [4], így a felhasználónak kell kezelnie az ezzel járó műveleteket (pl. konténer indítása, futtatása). A Launcher egységesíti a két megközelítést, így a felhasználónak most már csak egyetlen Python-programot kell elindítania. Ez a program a `huspacy` könyvtár felhasználásával elvégzi a HuSpaCy lefuttatását, valamint a `docker` programkönyvtár [5] felhasználásával nyit egy konténert, lefuttatja az e-magyart, majd a konténerben keletkezett elemzési eredményt átemeli a konténerből a lokális tárhelyre, végül a konténert leállítja és törli.

3.2. Logikai egységesítés

Az összehasonlításhoz elengedhetetlen, hogy a két elemző kimenetét egy egységes logika szerint lehessen értelmezni. Formailag mindkét elemző lehetővé teszi egy-egy elemzési táblázat létrehozását, amelynek soraiban a kapott tokenek,

[3] https://github.com/huspacy/huspacy, https://spacy.io/
[4] https://github.com/nytud/emtsv
[5] https://docker-py.readthedocs.io/en/stable/

oszlopaiban pedig az adott tokenre vonatkozó, elemzési szempontok szerinti kimenet látható. A megjelenítendő oszlopokat az alapján választottuk ki a HuSpaCy eszközkészletéből, hogy melyek azok a nyelvi szintek, amelyeken az elemzés megfeleltethető az e-magyar valamelyik elemzési szempontjának (token, lemma, morfológiai elemzések több típusban, szófaji címke, függőségi címke és fej).

A táblázatos forma létrehozása megkövetelt egyfajta elvi szintű egységesítést is, ugyanis a két elemző tokenizálási szemlélete is eltér, illetve a gyakorlatban létrejött eredmény is különbözhet több esetben (ez utóbbiról részletesebben ld. Eredmények szekcióban). A szemléletbeli eltérés kritikus pontja a *whitespace* karakterek kezelése. Az e-magyar tokenizálása viszonylag intuitív: a tokenek azok az egységek, amelyeket az elemzőrendszer egy szónak vagy egy központozási nyelvi formának ítélt. *Whitespace* karakterek alapvetően nem kerülnek a tokenek közé, csak azokban az esetekben, amikor azok is a token szerves részét képezik (pl. szóközzel tagolt számok esetén; „1 441-4000” egy tokennek számít). Ezzel szemben a HuSpaCy a *newline* karaktert tartalmazó *whitespace*-karakter-csoportokat tokenként tünteti fel, sőt, elemzést is próbál társítani hozzájuk, amelyeknek az esetek többségében természetesen nincsen értelme. Ezek a *whitespace* bemenetek szerencsére informatikailag kiszűrhetők, ezt a Launcher meg is valósítja.

A függőségi elemzés összehasonlításánál nagy különbséget okoz az, hogy nem ugyanazzal a címkehalmazzal dolgozik a két elemző, így ezek között konverzió szükséges. Az alkalmazás komponensei között így megtalálható a `dep_converter` fájlban egy egyszerű, szótár jellegű konverter, amely erre tesz kísérletet. Mivel a HuSpaCy által használt Universal Dependencies [6] címkehalmaz jóval bővebb az e-magyar által használt halmaznál (Vincze és mtsai, 2010), a konverzió a HuSpaCy címkéiről történik az e-magyar címkéire. A nem egyező számosság miatt a megfeleltetésben alapvetően történik információvesztés, és emellett néhány címke esetén éppen az e-magyar dolgozik részletesebb megnevezésekkel, így nem is teljesen egyértelmű a konverzió még egy irányban sem. Mindenesetre a konverter – bár elkerülhetetlenül magában hordoz egyfajta pontatlanságot – legalább alapvető szinten közös nevezőre tudja hozni a címkéket, így már értelmezhető közöttük összehasonlítás, egyenlőségvizsgálat.

A formai különbözőség mellett a dependencia elvi szinten is eltér az elemzőkben. Az e-magyar függőségi elemzése minden mondatban beszámozza a tokeneket 1-től a mondat hosszáig. A dependenciafa gyökerének választott elem a 0 fejcímkét kapja, a többi token pedig annak az elemnek az indexét, ami az ő feje a függőségi fában. 0 címkét kap még tipikusan a mondatzáró írásjel, illetve néhány nemtipikus esetben egy-egy olyan elem, amelyet az elemzés sehova sem tudott megfelelően bekötni (ld. Eredmények szekció). Így az e-magyar tisztán indexekkel dolgozik, amelyek közül néhányhoz (a 0-ás indexekhez) nem is tartozik token. Ehhez képest a HuSpaCy dependenciafejként rögtön azt a tokent adja vissza, amely fejként szolgál, ROOT esetében pedig önmagát. Így a Launcher a HuSpaCy logikájához igazodva visszavezeti az e-magyar indexeit is tokenekre, hogy lehessen tényleges egyenlőségvizsgálatot végezni a dependenciafejekre is.

[6] https://universaldependencies.org/u/dep/index.html

A névelemfelismerés esetében is szükség van egy minimális egységesítésre: az e-magyar által használt „1-” és „E-” címkéket a HuSpaCy „B-”, illetve „I-” címkéjére konvertáljuk.

A tokenizálásban azokat az eseteket is tudnunk kell kezelni, amikor a két elemző különböző tokenizálási eredményt ad. Ez általában elcsúszást eredményez az összehasonlítási táblázat soraiban. A Launcher ezt heurisztika segítségével szintén megoldja.[7]

3.3. Teljesítményjavítás

A morfológiai elemzéshez az e-magyarban 2 formátum áll rendelkezésre: az xpostag és az upostag (utóbbi jelenti a Universal Dependencies stílusú annotációt). Az elemzőláncben a morfológiai részt az EmMorph komponens valósítja meg. A HuSpaCy-ben szintén megtalálható egy integráción keresztül az EmMorph komponens, így a HuSpaCy is képes azt a részletes és szemléletes címkehalmazt használni, amelyet az e-magyar xpostag. A HuSpaCy először lefuttatja az EmMorph-komponens elemzését, majd ezt átkonvertálja UD-formátumra is (Vadász és Simon, 2019). Ezután lefuttatja a SpaCy-jellegű, saját morfológiai elemzését is, amely alapból UD-formátumú. Az integrációs feladat nehézsége abban rejlik, hogy az EmMorphból közvetlenül és konverzióval kapott eredmények listák a lehetséges elemzésekről, az integrált komponens ugyanis az egyértelműsítési logikát (tehát a valóban helyes elemzés kiválasztását) nem tartalmazza. Ezt kell megvalósítania az integráló kódrészletnek. Ehhez összeveti a lista elemeit a saját elemzésével, és az annak leginkább megfelelő EmMorph-elemzéseket adja vissza. Az integráció bizonyos szintű hibát visz a rendszerbe, ugyanis a kiválasztás alapjául szolgáló metrika sok esetben nem a helyes választást adja meg, illetve bizonyos tokenekre az integrált komponens sem fut le megfelelően (None eredményt kapunk). További érdekesség, hogy az eredeti integrációs változat használatával az elemzés több esetben össze is omlik, ugyanis nem tudja megfelelően kezelni a felső- és túlzófokú mellékneveket. Egy párhuzamosan futó projekt keretében ezt az integrációt informatikailag is javítottuk. Jelen összehasonlítást lefuttattuk az eredeti verzió[8] hibamentesített változatával[9] és a javított változattal is. Jelen tanulmányban a két változat eredményeit egyaránt közöljük, a javítási folyamat informatikai részleteit és a jövőben várhatóan elkészülő végleges verzióját egy külön tanulmányban mutatjuk be.

A Launcher lefuttatása után kapott eredményeket részletesen kiértékeltük, az ebből levont következtetéseket nyelvi szintenként közöljük.

[7] Ennek részletes leírását a Launcher dokumentációja tartalmazza.

[8] https://github.com/huspacy/huspacy/blob/master/docs/recipes/emmorph.md

[9] A továbbiakban ezt a hibamentesített változatot hívjuk eredeti változatnak, ugyanis a program logikája a hibajavítás mellett is alapvetően változatlan maradt.

4. Eredmények

4.1. Tokenizálás

A tokenszám alapvetően különbözik: a HuSpaCy 46607 tokent talált, míg az e-magyar 46452-t. Az eltérést kategorizálni lehet az alábbi típusokba:

- zárójelben („()" vagy „[]") lévő mondat utolsó szava utáni pontot az e-magyar a szó részének veszi (nem külön token a pont), a HuSpaCy pedig nem (153 előfordulás)
- „(1)" : HuSpaCy szerint 3 token, e-magyar szerint 2: „(" és „1)" (7 előfordulás)
- hármas kötőjeles szó: e-magyar szerint a kötőjelek és a szavak is külön tokenek (5 token), HuSpaCy szerint pedig az egész 1 db token (5 esetért felel)
- szóközzel tagolt számok: HuSpaCy külön tokennek veszi őket szóköz szerint, az e-magyar viszont egyben kezeli a számokat, 1 db tokent készít (5 esetért felel)
- sorszám (pl. „13.", „2022."): e-magyar szerint külön token a pont, HuSpaCy szerint nem (3 esetért felel)

A maradék elcsúszásért 1 vagy 2 előfordulásban egy-egy speciális token felel. Ezek szinte mind atipikus kifejezések, például rövidítések, speciális sorszámok, központozási csoportok, helyesírási hibás alakok közvetlen környezetében található kifejezések. Nem látható egyértelmű tendencia arra nézve, hogy az egyik elemző ezeket inkább egy tokennek, míg a másik többnek venné, vegyesen előfordul mindkét elemzőnél mindkét hozzáállás. Ha ezeket az eseteket is külön-külön számoljuk, összesen 15 szempont van, ahol a két elemző eltér.

Az eltérő eseteket manuálisan is áttekintettük és osztályoztuk aszerint, hogy biztosan hibás-e az egyik elemző, vagy pedig vitatható, hogy melyik elemzőnek adjunk igazat[10]. Az eset előfordulási számával való súlyozás nélkül a HuSpaCy 6 esetben egyértelműen jobb az e-magyarnál, az e-magyar 5 esetben a HuSpaCy-nél, és 4 db véleményes eset van. Így a HuSpaCy jobb az összes eset 40%-ában és az egyértelmű esetek 54,54%-ában. Súlyozással (188 hibahellyel és a fent közölt előfordulási számokkal) számolva a HuSpaCy 169 db esetben jobb, az e-magyar 11 esetben, illetve 8 véleményes esetről beszélünk. A HuSpaCy itt egyértelműen jobb, az összes eset 89,9%-ában győz az e-magyar elemzéséhez képest, az egyértelmű eseteket tekintve pedig ez az arány 93,9%-os. A zárójelben lévő mondatokon tehát az e-magyar szinte konzekvensen hibázik, és mivel a tesztszöveg ilyen esetekben bővelkedett, a 153 előfordulás a többi eltérési esethez képest kiugróan magas. Azonban súlyozás nékül is látható, hogy a HuSpaCy jobb teljesítményt nyújt, mint az e-magyar.

4.2. Morfológia

Ahogyan a Teljesítményjavítás részben is láthattuk, a morfológiai elemzéshez többféle kimenettel is dolgozhatunk. Jelen elemzésünkben az EmMorph komponens által használt címkehalmazzal dolgozunk, amely szemléletes és részletes

[10] Ez az elemzés, illetve minden további elemzés részletes adatai a GitHub repositoryban lesznek elérhetőek.

elemzést nyújt a tokenek morfológiai szerkezetéről. Így az összehasonlítás során az EmMorph-beli elemzési eredményeket hasonlítunk össze, amelyeket a két elemző kiválaszt mint helyes morfológiai elemzést. Az eredeti Huspacy-EmMorph integráció felhasználásával 46635 tokenen futtatva azt kaptuk, hogy 18796 esetben különbözött az elemzés (40,3%), speciális karakterek nélkül 11091 esetben (23,8%), és ebből azok az esetek, amikor a Huspacy-EmMorph None eredménnyel tért vissza (az integráció valamely részén történt probléma miatt) 4987-en voltak. A javított integrációval ezek a számok így változtak: 13883 különbség (29,8%), ebből a speciális karaktereket kiszűrve 6178 db (13,24%). (A None eredményű esetek száma nem változott.)

A speciális karakterek felelnek a HuSpaCy-EmMorphban a None kimenetek jelentős részéért, így ezeket leszűrve sokkal informatívabb képet kapunk arról, hogy milyen típusú szavakra nem működik megfelelően a HuSpaCy. Ezekről elmondható, hogy többnyire jól meghatározható típusú szavak: vezetéknevek, szervezetek nevei, helyesírási hibás alakok, számokat tartalmazó kifejezések. A helyes, csak valamiért az elemző számára szokatlan szavak közül néhány példa: *alacsonypadlós*, *ambiciózus*, *Covid* szót tartalmazó kötőjeles szóösszetételek stb.

Mivel a None kimenetek száma jelentős és az eltérések száma is viszonylag nagy, a HuSpaCy-EmMorph elemző több bizonytalanságot mutat az e-magyarnál. Ennek valamekkora része valószínűleg az integráció informatikai nehézségein alapszik, így a HuSPaCy-EmMorph a két eszköz közül jelenleg a kevésbé biztonságos választás.

4.3. Lemmatizálás

A lemmatizálás során is használhatjuk a HuSpaCy-EmMorph integrációt, amely elkészíti a lemmatizált alakot. Elemzésünkben az e-magyar saját lemmatizálóját, a HuSpaCy-EmMorph lemmatizálási eredményét, valamint a HuSpaCy SpaCy-féle lemmatizálóját használjuk. Ezzel 3 címke között vizsgálunk egyenlőséget.

Az eredeti integrációt használva 14826 különbséget kapunk (31,8%), speciális karakterek nélkül 7121-et (15,2%), amelyből 4987 None kimenet. A javított integrációval a különbségek száma 14398-ra csökken (30,9%), speciális karakterek nélkül számolva 6693-ra (14,4%), amelyből – a morfológiához hasonlóan – 4987 None kimenetű alak. Megfigyelhető tendencia, hogy az e-magyar és a HuSpaCy-EmMorph meglehetősen gyakran köznevesíti azokat a tulajdonnévi lemmákat, amelyekben köznevet ismer fel (így például a keresztneveket nem, a minisztériumok neveit viszont igen), és ezt nem feltétlenül egymással összhangban teszik. Viszont az e-magyar a csupa nagybetűs vezeték- és keresztneveket képes visszaalakítani megfelelő tulajdonnévi formára (pl. *BRENNER*-ből *Brenner*), míg a HuSpaCy ezeket meghagyja eredeti formájukban. Azonban az egyik megfigyelési szűrés során fény derült az e-magyar egy súlyosabb hibájára: a *Novák* vezetéknév lemmája 8 alkalommal *Nova* az e-magyar elemzésében.

További érdekesség, hogy a *-va/-ve* ragot a HuSpaCy meglehetősen nehezen tudja kezelni. Egy eset kivételével az összes olyan esetben, ahol hibázik - azaz vagy tévesen leválaszt egy -va/-ve szóvégződést, vagy pedig nem választ le egy -va/-ve toldalékot -, a lemma részének értékeli a *-va/-ve* toldalékot, abban az

egy kivételes esetben pedig rosszul képez lemmát a szóból (*vezényelve – vezényelv*). A *-va/-ve* végződésű szavakat leszűrve a ténylegesen ragozott alakokra 125 előfordulást kapunk (tegyük fel, hogy most csak a toldalék jelenlétét figyeljük, nem pedig ennek a helyességét). Ebből az *illetve* szónál konzekvensen egyik elemző sem választja le a toldalékot, és 9 további szó esetében szintén mindkét elemző a lemmán hagyja a *-va/-ve* ragot (ezeknél viszont nem minden esetben mondható el a konzekvencia, vagy túl kevés az előfordulás, hogy konzekvenciát tudjunk megállapítani). A HuSpaCy hasonlóan a lemma részének értékeli gyakran a *-hat/-het* képzőt is.

Mindhárom elemzőeszköz számára problémás a létige és annak különböző módokon képzett és ragozott alakjai (629 token, a viszonylag távolról származtatható alakokat is beleszámítva, pl. *lehetőség*). A HuSpaCy a múlt idejű és a jelen idejű alak különböző változataira mond *van* lemmát (320 eset), a jövő idejű létigéből származtatható igei és igenévi alakokat *lesz* lemmával jelöli (amennyiben felismeri benne a létigei tövet: *lehet*, *lenne*, *lennie* stb.; 201 előfordulás). Az e-magyar a *lesz* alakot a *lenne*, *lennie* stb. szóalakokra javasolja (100 eset). Az e-magyarnak *van* a válasza a *van*, *volt* és *lehet* alakjaira (436 eset). Egy *volt* előfordulásnál az e-magyar *volt* lemmát mond, a HuSpaCy *van*-t.

A None kimenetek nagy száma miatt kijelenthetjük, hogy a HuSPaCy-EmMorph rossz teljesítményt nyújt. A fent kiemelt eseteket tekintve az a konklúzió vonható le, hogy az e-magyar pontosabban dogozik. Amikor azonban hibázik, akkor a hibája súlyosabb, mint a HuSpaCy-é, amely bizonytalanság esetén többnyire változatlanul hagyja a szövegbeli alakot.

4.4. Szófaji címkézés

Összesen 4-féle mód van szófaji címkézésre: a HuSpaCy `pos` és `tag` elemzése, amely a SpaCy-ből ered, és egyszerű / részletezett címkézést hajt végre, valamint a HuSpaCy-EmMorph-ból eredő pos-címke és az e-magyar saját címkéi.

Az eredeti integrációval itt 13540 különbségről indulunk (29%), speciális karakterek nélkül 5835-ről (12,5%), amelyekből None kimenetű eset továbbra is 4987 (10,7%). A javított változat itt nem változtat a számokon.

Érdekes jelenség, hogy a két elemző között nincsen 100%-os egyetértés a PUNCT címkék kiosztásában. Az e-magyar az aposztrófot, valamint a megkülönböztetetten alsó és felső idézőjelet felismeri PUNCT elemként, a HuSpaCy viszont mindenféle más címkét ad ezeknek, PUNCT címkét egyszer sem. Olyan tokenek is előfordulnak, amelyeket a HuSpaCy PUNCT címkével látott el, az e-magyar pedig az ADJ, NOUN vagy NUM címkék valamelyikével: *):*, *§*, *15:00* stb. Mindkét elemzőnek vannak nehézségei a központozási karakterek megállapításában, viszont inkább az e-magyar kerül ki győztesként ebből az összehasonlításból.

Az eredeti bemenet első 929 tokenes részében kézzel is áttekintettük azt a 16 tokent, amelyeknél az elemzők különböző, de értelmes eredményt adtak (tehát azoktól az esetektől eltekintettünk, amelyeknek a HuSpaCy-EmMorph None kimenetet ad, hiszen ezek mindenképpen hamis eredményt adnak az összehasonlításnál). A 4 eszköz közül a HuSpaCy `tag` nyújtja a leggyengébb teljesítményt, a többi három nagyjából hasonlóan jól teljesít.

4.5. Függőségi elemzés

A függőségi elemzés kérdésében mindkét elemző elég bizonytalan. A kezdeti tesztek lefuttatása után úgy döntöttünk, a szöveg első néhány bekezdésén készítjük el az összehasonlítási elemzést, ugyanis ez egy mindössze 929 tokenes, belátható és manuálisan ellenőrizhető méretű bemenet. Ez a szövegrészlet nem tartalmazza azt a mondatot, amely az e-magyar elemzésében súlyos elvi hibát okoz.

Az elvi hiba az, hogy az e-magyar nem állapít meg ROOT címkét a következő mondatban: „(Szórványos taps a Momentum soraiban.)". (A mondat ténylegesen zárójelben van az eredeti szövegben.) Az a token, amelynek ezt a címkét fel kellene vennie (*taps*), APPEND címkét kap. A mondat szinte minden tekintetben normális, egy tipikus tokenizálási hiba van benne: a zárójelben lévő mondat utolsó szava és a mondatvégi írásjel egy tokenné kategorizálása. Azonban van egy minimálpárja és 94 hasonló szerkezetű („(Taps a [...] soraiban.)") rokon jellegű mondata a szövegben, és mindegyik kap ROOT címkét.

Elvi hibája a HuSPaCy-nek is van; erről esett már néhány szó a különbségek kiegyelésénél. Mivel a dependenciafej választása a teljes tokenkészletből dolgozik, előfordul, hogy egy *whitespace*-csoportot kap fejként valamelyik token.[11]

A dependenciacímke 289 esetben (31,1%) különbözik a két elemző esetében (itt megjegyezzük, hogy ennek bizonyos részét a konverzióból adódó hibák is okozhatják). Ebből azon esetek száma, ahol a fej sem egyezik, 222 db (23,9%). A fej összesen 437 helyen (47%) nem egyezik a két elemzőnél. Az adatokra ránézve itt is láthatjuk, hogy a speciális karakterek meglehetősen kérdéses esetek, így itt is kiszűrhetjük őket. Ekkor a dependenciacímke 286 (30,1%) esetben különbözik, a dependenciafej 306 esetben (32,9%), mindkettő egyszerre pedig 221 esetben (32,8%) tér el.

Érdekes eset az *és* kötőszó is. A 16 db előforduló *és* token mindegyike esetében különbözik a fej, és 15 esetében meg is állapítható, hogy az e-magyar az első lehetőséget, a HuSpaCy pedig a második lehetőséget választja. Ebben a tekintetben tehát a HuSpaCy működik az UD elveknek megfelelően[12].

[11] Ezt azért nem lehet megoldani ugyanolyan eredményszintű szűréssel, mint az eddigieket, mert itt „bemenetet" kellene szűrnünk, azt a halmazt, amiből a HuSpaCy dolgozik a fejek választásánál. Kisebb technikai nehézség, hogy amikor a fejként kapott *whitespace*-csoport tartalmaz sortörést, akkor ez ténylegesen sortörésként (nem pedig „\n" karakterként) jelenik meg, így elcsúsztatja a táblázatot. Ezt is megoldja a Launcher olyan módon, hogy az ilyen esetekben a valódi fej helyett „HEAD IS WHITESPACE!" üzenettel helyreállítja a táblázatot. Mivel a bemeneti szöveg gyakorlatilag bármilyen mennyiségű és összetételű *whitespace*-csoportot tartalmazhat, egységesen kezeljük a sortörés tartalmazásának ténye alapján az összes lehetőséget. (A szóközök és tabulátorok nem okoznak ilyen elcsúszást, így ezeket eredeti formájukban hagyjuk, így is megfelelő információt hordoznak.) Ezzel veszítünk annyi információt, hogy pontosan milyen összetételű a csoport, de mivel egyébként is értelmetlen a token, és nyilván hibás, hogy fejként szerepel, ezzel a gyakorlatban nem veszítünk érdemi információt. A rövidített bemeneti fájlban 8 ilyen eset szerepel (0,9%).

[12] A Universal Dependencies útmutatása alapján a mellérendelések első elemét tekintjük a szülő csomópontnak: „Coordinate structures are in principle symmetri-

Összességében tehát azt látjuk, hogy az e-magyar van, hogy eltér a Universal Dependencies formátumtól, amelyre az elemzőt használó jogosan számíthatna. Függőségi elemzésre inkább a HuSpaCy használata alkalmas, kiszámíthatósága és bővebb címkehalmaza miatt.

4.6. Névelem-felismerés

A HuSpaCy egyértelműen sokkal több névelemet talál meg egy adott szövegben, mint az e-magyar. Ennek feloldásához az alap országgyűlési fájl első néhány bekezdését (929 tokenes rész) manuálisan is annotáltuk, és ehhez viszonyítva is megnéztük az elemzők teljesítményét.

A névelemek felismerését többféleképpen is megközelíthetjük, és az alább felsoroltak közül a Launcher mindegyiket megvalósítja. Az első megközelítés az, hogy minden egyes tokenről meg tudjuk mondani, hogy névelem-e, ehhez használhatjuk az IOB-notációt, amely mindkét elemzőben megtalálható. Az eredmények a teljes fájlon futtatva: 46635 tokenből 1402 esetben (3%) van eltérés. A HuSpaCy 44099 db tokent azonosított nem-névelemként, az e-magyar pedig 44689 db-ot, ebből az 590-es különbségből látható, hogy 10000-es nagyságrendű hosszú bemenet esetén körülbelül 100-as nagyságrendű eltérés van az elemzők között. A kézzel annotált, rövidített tesztfájlon 929 tokenből 18 különbséget látunk (1,9%), a HuSpaCy 10-zel több névelemet talált.

A másik megközelítés arra fókuszál, hogy egy adott tulajdonnév milyen címkét (címkéket) kap a teljes szövegen belül. A Launcher által előállított táblázatban elsőként a HUSpaCy és e-magyar által megítélt címkék halmazainak egyenlőségét látjuk, majd a közösen megtalált névelemet, illetve rendre a HuSpaCy és az e-magyar által megítélt címkék halmazát. Természetesen vannak névelemek, amelyeket a másik elemző nem feltétlenül talált meg, ezeket a táblázat után listába gyűjtve láthatjuk (ezúttal az egyenlőségvizsgálat és a másik elemző halmaza nélkül). Ez a típusú megközelítés 283 közösen megtalált névelemet jelez az eredeti bemeneten, valamint 288 csak a HuSpaCy által megtalált névelemet és 28 csak az e-magyar által megtalált névelemet. (Fontos megjegyezni, hogy itt egy token teljesen megegyező előfordulásait egynek számoljuk.) A 284 db közösen megtalált névelem esetén a címkehalmaz 51 esetben tér el (18%).

A címkéket összefoglalva az látszik, hogy az e-magyar esetében az ORG megállapítása a legnehezebb. Az e-magyar által ORG-nak ítélt szavak szinte mindegyikéhez a HuSpaCy talált más értelmezést is. Az önmagukban álló vagy toldalékolt országnevekben konzekvens eltérés figyelhető meg: a HuSpaCy mindig ORG-nak, az e-magyar mindig LOC-nak tekinti őket. Az e-magyar sok esetben nyilvánvalóan téved (pl. személyneveknél ezt a szöveg vagy *gold standard* címkézés nélkül is meg lehet állapítani), így erősen valószínű, hogy a HuSpaCy teljesítményét ebben a komponensben jobbnak mondhatjuk. Az 4.6 táblázatban láthatunk egy összehasonlítást a rövid fájlrészleten, manuális elemzéssel.

cal, but the first conjunction is by convention treated as the parent (or "technical head") of all subsequent coordinated clauses via the conj relation." (`https://`

névelem	kézi	HuSpaCy	e-magyar	névelem	kézi	HuSpaCy	e-magyar
Országgyűlés	ORG	ORG	ORG	Szűcs Lajos	PER	PER	PER
Hiszékeny Dezső	PER	PER	PER	Ház	ORG	nincs	ORG
Országgyűlés(ről)	ORG	ORG	nincs	Országgyűlés	ORG	ORG	ORG
Fidesz	ORG	ORG	ORG	Országgyűlés	ORG	MISC	ORG
Ungár Péter	PER	PER	PER	LMP	ORG	ORG	nincs
UNGÁR PÉTER	PER	ORG	nincs	LMP	ORG	ORG	nincs
Országgyűlés	ORG	PER	ORG	EU(-integráció)	ORG	nincs	nincs
Magyarország	LOC	LOC	LOC	Magyarország(on)	LOC	LOC	LOC
Thomas Piketty	PER	PER	PER	Magyarország	LOC	LOC	LOC
Európai Unió(tól)	ORG	ORG	ORG	Magyarország	LOC	ORG	LOC
Európai Unió	ORG	ORG	ORG	Európai Unió(t)	ORG	ORG	ORG
Egyesült Államok(kal)	LOC	ORG	LOC	Európai Unió	ORG	ORG	ORG
Audi(nak)	ORG	ORG	ORG	Tanács	ORG	ORG	nincs
Rétvári Bence	PER	PER	nincs	Audi	ORG	ORG	ORG
Magyarország(nak)	LOC	ORG	LOC	Audi(nak)	ORG	ORG	nincs
Audi(nak)	ORG	ORG	ORG	Ház(nak)	ORG	ORG	ORG
Bundestag(nak)	ORG	ORG	ORG				

1. táblázat. NER-összehasonlítás. Az első és ötödik oszlop a humán elemzés során megtalált névelemeket, a „kézi” oszlop a humán címkét, a HuSpaCy és e-magyar oszlopok pedig az elemzők által adott címkét tartalmazzák, illetve a „nincs” szót, ha az adott elemző az adott névelemet nem jelölte meg.

A manuális elemzés 35 db néveléméből teljes egyezés 19 db (54,3%). A HuSpaCy 25 alkalommal címkézett jól (71,43%), 2 alkalommal (5,71%) nem találta meg a névelemet, 7 alkalommal rossz címkét adott (20%). Az e-magyar 26 alkalommal teljesített jól (74,29%), 8 névelemet nem talált meg (22,86%), nem volt olyan, hogy rossz címkét adott volna (0%).

Összességében megállapíthatjuk, hogy a HuSpaCy jóval nagyobb fedéssel dolgozik, viszont az e-magyar típuscímkéi pontosabbak.

5. Összegzés

Az összehasonlítás eredményei alapján megállapítható, hogy a HuSpaCy és az e-magyar elemzők különböző erősségekkel és gyengeségekkel rendelkeznek, így a választás a konkrét alkalmazási céltól függ. A HuSpaCy kiváló teljesítményt nyújt tokenizálásban, névelem-felismerésben és dependenciacímkézésben, különösen ipari környezetben, ahol gyorsasága és precizitása előnyt jelenthet. Ugyanakkor a morfológiai és lemmatizálási teljesítménye, különösen az EmMorph integráció során tapasztalt problémák miatt, alulmúlta az e-magyart. Az e-magyar elemző pontosabb morfológiai és lemmatizálási eredményeket nyújtott. Összehasonlításunk természetesen nem teljeskörű; egy adott stílusrétegbe tartozó szöveg

`universaldependencies.org/u/dep/conj.html`), a kötőszót pedig mindig az utolsó elemhez kötjük (`https://universaldependencies.org/u/dep/cc.html`).

egy példányán vetettük össze a két elemzőt, korlátozottan kvantitatív módon. Teljesebb, pontosabb és nagyobb volumenű összehasonlítást tenne lehetővé egy olyan szöveg, amely megfelelően nagy méretű, és minden nyelvi szinten *gold standard* annotációval van ellátva. Ennek előállítása még várat magára. További feladataink közé tartozik a HuSpaCy-EmMorph integráció javítása, hogy az egyébként kiegyensúlyozott teljesítményt nyújtó HuSpaCy a lemmatizálás és a morfológia terén az e-magyaréhoz hasonló pontossággal tudjon dolgozni.

Köszönetnyilvánítás

A kutatást az MTA „Tudomány a Magyar Nyelvért Nemzeti Program" támogatta.

Hivatkozások

Csendes, D., Csirik, J., Gyimóthy, T.: The Szeged Corpus: A POS tagged and syntactically annotated Hungarian natural language corpus. In: Proceedings of the 5th International Workshop on Linguistically Interpreted Corpora (LINC 2004) at The 20th International Conference on Computational Linguistics (COLING 2004). pp. 19–23 (2004)

Honnibal, M., Montani, I., Van Landeghem, S., Boyd, A.: spaCy: Industrial-strength Natural Language Processing in Python (2020)

Indig, B., Sass, B., Simon, E., Mittelholcz, I., Kundráth, P., Vadász, N.: emtsv – Egy formátum mind felett. In: Magyar Számítógépes Nyelvészeti Konferencia. Szeged (2019), in Hungarian

Novák, A.: A New Form of Humor – Mapping Constraint-Based Computational Morphologies to a Finite-State Representation. In: Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC 2014). European Language Resources Association (ELRA), Paris (2014)

Novák, A., Siklósi, B., Oravecz, Cs.: A new integrated open-source morphological analyzer for hungarian. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016). European Language Resources Association (ELRA), Paris (2016)

Orosz, Gy., Novák, A.: PurePos 2.0: A Hybrid Tool for Morphological Disambiguation. In: Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2013). pp. 539–545 (2013)

Orosz, Gy., Szabó, G., Berkecz, P., Szántó, Zs., Farkas, R.: Advancing Hungarian Text Processing with HuSpaCy: Efficient and Accurate NLP Pipelines. In: Ekštein, Kamil and Pártl, František and Konopík, Miloslav (szerk.) Text, Speech, and Dialogue. pp. 58–69. Springer Nature Switzerland, Cham (2023)

Orosz, Gy., Szántó, Zs., Berkecz, P., Szabó, G., Farkas, R.: HuSpaCy: an industrial-strength Hungarian natural language processing toolkit. In: XVIII. Magyar Számítógépes Nyelvészeti Konferencia. pp. 59–73 (2022)

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)

Simon, E., Indig, B., Kalivoda, A., Mittelholcz, I., Sass, B., Vadász, N.: Újabb fejlemények az `e-magyar` háza táján. In: Berend, G., Gosztolya, G., Vincze, V. (szerk.) XVI. Magyar Számítógépes Nyelvészeti Konferencia. pp. 29–42. Szegedi Tudományegyetem Informatikai Tanszékcsoport, Szeged (2020)

Straka, M.: UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In: Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. pp. 197–207. Association for Computational Linguistics, Brussels, Belgium (October 2018), `https://www.aclweb.org/anthology/K18-2020`

Vadász, N., Simon, E.: Konverterek magyar morfológiai címkekészletek között. In: XV. Magyar Számítógépes Nyelvészeti Konferencia. pp. 99–111. Szegedi Tudományegyetem TTIK, Informatikai Intézet, Szeged (2019)

Vincze, V., Szauter, D., Almási, A., Móra, Gy., Alexin, Z., Csirik, J.: Hungarian Dependency Treebank. In: Proceedings of LREC 2010. ELRA, Valletta, Malta (May 2010)

Váradi, T., Simon, E., Sass, B., Gerőcs, M., Mittelholtz, I., Novák, A., Indig, B., Prószéky, G., Vincze, V.: Az e-magyar digitális nyelvfeldolgozó rendszer. In: XIII. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY2017). pp. 49–60. Szegedi Tudományegyetem Informatikai Tanszékcsoport, Szeged (2017)

Zsibrita, J., Vincze, V., Farkas, R.: magyarlanc: A toolkit for morphological and dependency parsing of Hungarian. In: Proceedings of RANLP. pp. 763–771 (2013)