Transformer-alapú HuSpaCy előelemző láncok

Szabó Gergő, Orosz György, Szántó Zsolt,
Berkecz Péter, Farkas Richárd

Szegedi Tudományegyetem, Informatikai Intézet
Szeged, Árpád tér 2.
{gszabo,szantozs,berkecz,rfarkas}@inf.u-szeged.hu
gyorgy@orosz.link

Kivonat Évről évre jelennek meg egyre pontosabb nyílt forráskódú természetes-nyelvfeldolgozó transformer-alapú nyelvi modellek. Viszont ezekre épülő, kifejezetten magyar nyelvre fejlesztett előelemző lánc, ami tartalmaz tokenizálást, mondatra bontást, lemmatizálást, szófaji egyértelműsítést, morfológiai címkézést, függőségi elemzést és névelem-felismerést, idáig nem létezett. Dolgozatunkban bemutatunk egy, a fent említett feladatokat megvalósító transformer-alapú előelemző láncot, amit több különböző magyarra elérhető nyelvi modellel is teszteltünk. Az elkészült rendszer a feladatok nagy részében state-of-the-art eredményeket ért el. Arra is kellő figyelmet fordítottunk, hogy megvizsgáljunk különböző erőforrás-igényű lehetőségeket és bemutassunk olyan módszereket, amelyekkel a leghatékonyabbnak bizonyult rendszerek memóriahasználata csökkenthető.
Kulcsszavak: transformer, huspacy, nyelvi előfeldolgozás

1. Bevezetés

Az elmúlt években nagyon elterjedtek a transformer-alapú modellek alkalmazása (például: `BERT` (Devlin és mtsai, 2018)). Ezek a mély neuronhálókat használó, nagy mennyiségű szövegen előtanított nyelvi modellek azonnal state-of-the-art eredményeket értek el a természetes-nyelvfeldolgozás különböző feladatain (Devlin és mtsai, 2018). Bár már magyar nyelvre is elérhetők ilyen modellek (Nemeskey, 2022) (Conneau és mtsai, 2019), olyan ezekre épülő, kifejezetten a magyar nyelv igényeit figyelembe vevő általános előelemző keretrendszer eddig nem készült, ami egy architektúraként foglalná össze a tokenizálást, a mondatra bontást, a szófaji egyértelműsítést, a morfológiai elemzést, a lemmatizálást, a függőségi elemzést és a névelem-felismerést. Célunk egy ilyen eszköz létrehozása volt.

Elemzőnk alapjául a spaCy (Honnibal, 2015) keretrendszer magyar nyelvű változatát, a HuSpaCy-t (Orosz és mtsai, 2022) választottuk. A HuSpaCy egy Python nyelvre építő, nyílt forráskódú, szabadon felhasználható nyelvfeldolgozási keretrendszer, ami a korábban felsorolt komponensek mindegyikét tartalmazza. Cikkünkben bemutatjuk, hogy a HuSpaCy-ben található szóbeágyazások lefedő konvolúciós rétegekre épülő neuronháló esetében, milyen javulás érhető el az

előtanított transformer-alapú modellek beépítésével. Kísérleteinkhez a kizárólag magyar adaton tanított `huBERT`-et (Nemeskey, 2022) és két többnyelvű modellt, a Multilingual BERT-et (Pires és mtsai, 2019) és az `XLM-RoBERTa`-t (Conneau és mtsai, 2019) használtuk.

A transformer modellek hátránya a nagy számításikapacitás- és a memóriaigényük. Utóbbi kifejezetten igaz a többnyelvű modellek esetén, hiszen ezen modellek szótára óriási mennyiségben tartalmaz olyan szótöredékeket, amik a magyar nyelvben nem fordulhatnak elő. Ennek a problémának a kezelésére megszűrtük az `XLM-RoBERTa` szótárát és megvizsgáltuk, hogy ez milyen hatással van a rendszer teljesítményére.

A további hatékonyság javítás érdekében a HuSpaCy címkézésért felelős neurális architektúrájában két ponton változtattunk. A korábban használt átmenet-alapú függőségi elemzőt kicseréltük egy gráf-alapú változatra, ezen felül a névelem-felismerőt is bővítettük a beam search (Medress és mtsai, 1977) beépítésével.

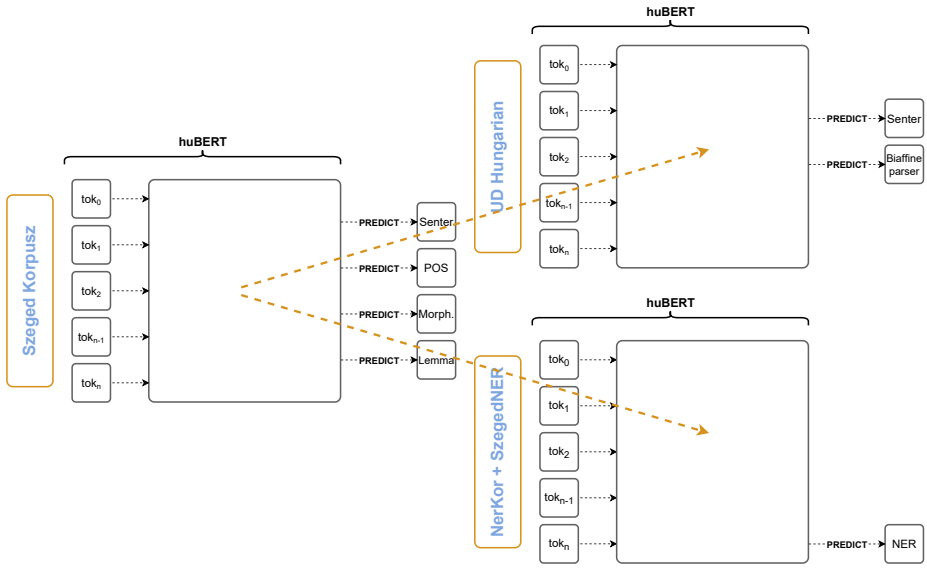
2. Kapcsolódó munkák

Kutatásunk fő célja, hogy megvizsgáljuk, hogy a legújabb transformer-alapú megoldások hogyan teljesítenek a korábbi megoldásokkal szemben. A magyar nyelvre már korábban is fejlesztettek teljes előelemző láncokat. Ilyen például a `HuSpaCy-large` (Orosz és mtsai, 2022) modellje, amely a WebCorpus 2.0-án (Nemeskey, 2020b) tanított statikus szóbeágyazásokat használ, illetve egy 4 rétegű CNN-t (*Convolutional Neural Networks*); a `magyarlanc` (Zsibrita és mtsai, 2013), amely a rejtett Markov modell-alapú PurePos-t (Orosz és Novák, 2013) használja és egy gráf-alapú függőségi címkézőt; a `Stanza` (Qi és mtsai, 2020), amely LSTM-alapú (*Long Short Term Memory*) és amely magas pontszámokat ért el az UD korpuszok terén; az `UDPipe` (Straka, 2018), amelyet alaprendszerként használnak a CoNLL versenyeken; a `UDify` (Kondratyuk és Straka, 2019), ami egy többnyelvű `BERT`-alapú modell és az `emtsv` (Simon és mtsai, 2020; Indig és mtsai, 2019a,b; Váradi és mtsai, 2018, 2017), amely kifejezetten a magyar nyelv igényeire lett fejlesztve, illetve amibe bele lett integrálva az `emBERT` (Nemeskey, 2020a) transformer-alapú NER rendszere. A feladatok tekintetében a HuSpaCy és a `Stanza` képes elemezni az összes eddig taglalt feladatot. Ezen túl az `emtsv` is szintén, viszont a függőségi elemzője nem Universal Dependencies formátumú. Az `UDPipe`-ból és a `magyarlanc`-ból a névelem-felismerő maradt ki.

3. Rendszer felépítése

3.1. Alapmodell felépítése

Rendszerünk alapja a HuSpaCy (Orosz és mtsai, 2022) cikkben bemutatott, többfeladatos tanulásra építő architektúra képezi, ami egy lépésben kezeli a különböző címkézési feladatokat. Az ebben szereplő 4 rétegű konvolúciós hálókra építő architektúrát lecseréltük egy előtanított transformer-alapú modellre.



1. ábra: Transfer learning ábrázolása.

Ahogy az 1. ábrán is látható, a rendszerünk 3 különálló modellből áll. Az első, Szeged Korpuszon (Csendes és mtsai, 2004) tanított modell végzi a szófaji egyértelműsítést, lemmatizálást és a morfológiai címkézést. Ezt a modellt lemásoltuk két példányban és továbbtanítottuk az egyiket függőségi elemzésre kiegészítve a mondatra bontás feladattal is, a másikat pedig névelem-felismerésre. Erre azért volt szükség, mert az egyes feladatok tanítására eltérő adatbázisok álltak rendelkezésünkre és így mégis képesek voltunk kihasználni az egyes részfeladatok közötti kapcsolatokat. Függőségi elemzésre a Szeged Korpusznál kisebb, viszont nemzetközileg elterjedt címkekészletet használó magyar Universal Dependencies (Nivre és mtsai, 2017) (UD) korpuszt használtuk. A névelem-felismerőt pedig a NerKor (Simon és Vadász, 2021) és a SzegedNER (Szarvas és mtsai, 2006a) összevont korpuszán tanítottuk.

Ez a módszer abban tér el a korábbi HuSpaCy architektúrától, hogy csak 2 modell készült. A Szeged Korpuszon tanult modell tovább volt tanítva az Universal Dependencies korpuszon, amely így már ki volt egészítve a függőségi elemző fejjel. Így nem volt egy külön modell a függőségi elemzésre. A változtatásra a transformer architektúra esetén azért volt szükség, mert a függőségi elemző tanítása közben azt tapasztaltuk, hogy a modell elfelejtette a korábban tanult feladatokat. A lemmatizálás, szófaji egyértelműsítés és morfológiai címkézés feladatok esetén a pontosság lezuhant az első néhány epochban, utána javult minimálisan, de például a morfológiai címkézés esetén 6, a lemmatizálás esetén pedig 27 százalékponttal romlott a pontosság a Szeged Korpuszon tanult modellhez képest.

3.2. Az alapmodell kiegészítései

A HuSpaCy rendszer által alapértelmezettként használt átmenet-alapú függőségi elemzőt lecseréltük a spaCy rendszer által biztosított gráf-alapú, biaffine figyelmi *(attention)* mechanizmussal rendelkező elemzőre (Dozat és Manning, 2016). Az eredeti cikkben alkalmazott Bi-LSTM rétegek helyett, ahogy a korábbi feladatokban, itt is transformer-alapú architektúrát használtunk. Az átmenet-alapú rendszerek a függőségi fát osztályozási problémák sorozataként építik fel, amikben lehetőség van két vizsgált szó között él behúzására vagy balról jobbra haladva a vizsgált szavak léptetésére. Ezek a megoldások a mohó, lokális döntéseik miatt rosszul tudnak teljesíteni hosszú mondatok esetén. Ezzel szemben a gráf-alapú elemzők a szavakból, mint csúcsokból felírt teljes gráfban keresik a maximális feszítőfát.

A névelem-felismerő a függőségi elemzőhöz hasonlóan egy átmenet-alapú megoldást alkalmaz, amit beam search használatával bővítettünk. Ennek lényege, hogy az elemző az egyes lépésekben több korábbi elemzést is figyelembe vesz. A lemmatizálás feladatban a Berkecz és mtsai (2023) cikkben használt megoldásokat alkalmaztuk.

A kifejezetten magyar adatokon tanított `huBERT` mellett kísérleteztünk más, többnyelvű modellekkel is, amelyek támogatják a magyar nyelvet is. Az egyik a `BERT-base-Multilingual` volt, illetve az `XLM-RoBERTa` két különböző méretű változata, az `XLM-RoBERTa-base` és az `XLM-RoBERTa-large`.

A `huBERT`, a `BERT-base-Multilingual` és az `XLM-RoBERTa-base` mindegyike 12 rétegből áll, amelyek egyenként 768 dimenziós vektorokat használnak. Az előbbi kettő összesen 110 millió paraméterrel rendelkezik, az `XLM-RoBERTa-base` pedig 125 millióval. Ezzel szemben az `XLM-RoBERTa-large` kétszer annyi réteget és 1024 dimenziós vektorokat használ és összesen 355 millió paraméterrel rendelkezik.

3.3. Többnyelvű modellek méretcsökkentése

A többnyelvű modellek egyik hátránya a nagyobb memóriaigényük, aminek az oka, hogy az eltérő nyelvek szavai miatt sokkal nagyobb szótárra van szükségük. Egynyelvű felhasználás esetén viszont a szótár egy jelentős részére szinte biztosan nincs szükségünk. Ilyenek például a kínai szimbólumok, török betűk, cirill betűk és az azokra épülő szavak, szótöredékek. Ennek kezelésére reguláris kifejezésekkel megszűrtük a két `XLM-RoBERTa` modell szótárát. A jelenlegi szűrés csak magyar nyelvben használt betűket és írásjeleket tartalmazó szótöredékeket hagyta meg.

4. Eredmények

Az összes kiértékelés (a névelem-felismerő kivételével) a Hungarian Universal Dependencies Corpus tesztkészletén (De Marneffe és mtsai, 2021) készült és a CoNLL 2018 Shared Taskon[1] alkalmazott kiértékelő szkript felhasználásával tör-

[1] `https://universaldependencies.org/conll18/conll18_ud_eval.py`

tént. A nem transformer-alapú modellek esetén összehasonlításképpen öt rendszert választottunk. Az első az `emtsv` (Simon és mtsai, 2020; Indig és mtsai, 2019a,b; Váradi és mtsai, 2018, 2017), második az `UDPipe` (Straka, 2018), a harmadik a `Stanza` (Qi és mtsai, 2020) és a negyedik a `UDify` (Kondratyuk és Straka, 2019). Ugyanakkor a 2021-ben publikált `HuSpaCy-large` (Orosz és mtsai, 2022) modelljével is összehasonlítottuk.

A `Stanza`[2] és `UDPipe`[3] szerzői már kimérték a rendszerüket az UD teszthalmazon, de ez nem mondható el az `emtsv`-ről. Itt a rendszert az alapértelmezett konfigurációs beállítások megtartásával lefuttattuk az általunk alkalmazott teszthalmazon. Ugyanakkor fontos kiemelni, hogy az `emtsv` eredményei olyan szempontból nem számítanak összevethetőnek, hogy nem lett újratanítva az általunk használt adatbázis vágásokkal, hanem csak egyszerű kiértékelés történt. Így a kiértékelés során előfordulhatott átfedés a tanító és a kiértékelő adathalmazok között. A `UDify` kiértékelése során gold adatot használt a tokenizálás és a mondatra bontás feladatokban.

A transformer-alapú modellek esetében felhasználtuk az eddigi tapasztalatainkat és további kísérleteket végeztünk néhány egyéb előtanított nyelvi modellel a HuSpaCy mostanra már alapértelmezett konfigurációjával. A következőkben részletesen bemutatjuk az eredményeket, valamint azt is, hogy mely nyelvi modellekkel sikerült elérni őket.

4.1. Tokenizálás és mondatra bontás

A mondatra bontás esetén látható, hogy minimálisan, de a transformer modellek meghaladják a nem transformer-alapú modelleket. Továbbá látható, hogy az `XLM-RoBERTa-large`-alapú modell szinte tökéletesen szegmentált, így megelőzve társait. A tokenizálás semmilyen technikában nem tért el a HuSpaCy cikkben ismertetett módszerektől.

	Token	Mondatra bontás
`Stanza`	**99,92%**	97,45%
`UDPipe`	99,80%	95,90%
`emtsv`	99,77%	98,67%
`UDify`	–	–
`HuSpaCy-large`	99,89%	98,55%
`huBERT`	99,89%	99,33%
`BERT-base-Multilingual`	99,89%	87,75%
`XLM-RoBERTa-base`	99,89%	99,33%
`XLM-RoBERTa-large`	99,89%	**99,67%**

1. táblázat. Mondatra bontás F1-score eredményei az UD teszthalmazán.

[2] `https://stanfordnlp.github.io/stanza/performance.html`
[3] `https://ufal.mff.cuni.cz/udpipe/1/models`

4.2. Morfo-szintaktikai elemzés

A 2. táblázat eredményei alapján elmondható, hogy a `huBERT`-alapú modell kiemelkedően teljesít minden területen. A morfológiai elemzésben sikerült több, mint fél százalékponttal javítani az eddigi legjobb eredményen. A szófaji címkézés esetében szintén sikerült javítani több, mint egy százalékponttal. Ezen felül a lemmatizálás feladatban is legalább 4 százalékponttal javított minden eddigi rendszerhez képest.

Ugyanakkor összesítve az `XLM-RoBERTa-large`-alapú modell ismét majdnem minden más modellt (lemmatizálásnál szinte fej fej mellett van a `huBERT`-alapú modellel) megelőz, morfológiai elemzés esetén is 3 százalékponttal felülmúlja a `huBERT`-alapú modellt.

	Lemma pontosság	PoS pontosság	Morf. pontosság
`Stanza`	94,19%	96,00%	93,62%
`UDPipe`	88,50%	90,60%	88,50%
`emtsv`	96,16%	89,19%	89,12%
`UDify`	90,19%	96,36%	86,16%
`HuSpaCy-large`	97,46%	96,89%	93,87%
`huBERT`	**98,68%**	97,27%	94,20%
`BERT-base-Multilingual`	98,48%	97,11%	91,76%
`XLM-RoBERTa-base`	98,55%	97,73%	96,61%
`XLM-RoBERTa-large`	98,67%	**98,01%**	**97,15%**

2. táblázat. Lemmatizálás, szófaji címkéző (PoS) és morfológiai elemző F1-score eredményei az UD teszthalmazán.

4.3. Gráf-alapú függőségi elemző

A 3. táblázatban látható az átmenet-alapú függőségi elemző, a gráf-alapú függőségi elemző, illetve a többi rendszer eredményei. Mivel az `emtsv` nem szolgáltat Universal Dependencies formátumú függőségi elemzést, így ez a modell nem mérhető ezzel a feladattal.

A meglévő rendszerek és a `HuSpaCy-large` modellje is alulmarad a `huBERT` átmenet-alapú modellel szemben, mind az UAS és a LAS tekintetében is. Az átmenet-alapú függőségi elemzővel rendelkező `huBERT`-alapú modell viszont alulmarad a már tárgyalt gráf-alapú megoldással szemben. Ezért ezt alkalmaztuk is a továbbiakban, illetve a többi transformer-alapú modellnél is ezt használtuk alapértelmezett elemzőként függőségi elemző esetén.

Viszont ebben a feladatban az `XLM-RoBERTa-large` pontatlanabb a `huBERT` gráf-alapú modellel szemben. Ez az egyetlen olyan komponens, ahol a `huBERT`-alapú modell sokkal jobban teljesít az `XLM-RoBERTa-large`-alapú modellnél.

	UAS	LAS
`Stanza`	84,19%	79,23%
`UDPipe`	72,80%	67,20%
`UDify`	89,69%	84,88%
`HuSpaCy-large`	82,53%	75,56%
`huBERT` - átmenet-alapú	89,95%	83,94%
`huBERT` - gráf-alapú	**91,57%**	**87,58%**
`BERT-base-Multilingual` - gráf-alapú	85,92%	81,93%
`XLM-RoBERTa-base` - gráf-alapú	89,47%	86,02%
`XLM-RoBERTa-large` - gráf-alapú	90,38%	86,88%

3. táblázat. Átmenet-alapú és gráf-alapú függőségi elemzés összehasonlítása a meglévő rendszerekkel, illetve a magyar nyelvre elérhető transformer-alapú nyelvi modellek kiértékelése.

4.4. Névelem-felismerés

Mivel az UD korpusz nem tartalmaz névelemeket, ezért erre a célra a NerKor (Simon és Vadász, 2021) és a SzegedNER (Szarvas és mtsai, 2006a) korpuszt használtuk. Mivel az `UDPipe`-ban nincs névelem-felismerő, így azt nem lehetett bevonni a vizsgálatba. Három korábbi névelem-felismerőt is bevettünk az összehasonlításba. Az egyik a Szarvas és mtsai (2006b), amely döntési fákat használ. A második a Simon (2013), amely egy lineáris modellt használ és rejtett Markov modelleket kombináló statisztikai címkézővel rendelkezik. A harmadik, újabb rendszer pedig az `emBERT`, amely egy transformer-alapú NER modell. A `Stanza` esetén viszont fontos kiemelni, hogy újra lett tanítva az általunk használt adatbázisokkal (Simon és mtsai, 2022).

	SzegedNER	NerKor	Kombinált
Simon (2013)	95,06%	–	–
Szarvas és mtsai (2006b)	94,77%	–	–
`emBERT`	97,40%	92,09%	**92,99%**
`Stanza`	91,78%	80,53%	83,75%
`HuSpaCy-large`	95,31%	80,75%	85,73%
`huBERT`	97,01%	88,27%	90,26%
`huBERT` + beam search	97,37%	89,13%	91,27%
`BERT-base-Multilingual` + beam search	–	–	89,07%
`XLM-RoBERTa-base` + beam search	–	–	90,94%
`XLM-RoBERTa-large` + beam search	–	–	91,86%

4. táblázat. F1-scoreon mért névelem-felismerés összehasonlítása a SzegedNER, a NerKor és a kombinált teszthalmazon is.

A NerKor adatbázis esetében a szerzők által meghatározott tanító-validáló-teszt halmazokra bontást alkalmaztuk. A SzegedNER esetében pedig, az összehasonlítás érdekében, a Szarvas és mtsai (2006b) cikk által meghatározott vágás lett felhasználva.

A meglévő rendszerek, az `emBERT` kivételével, alulmaradnak a `HuSpaCy-large` modellel és a `huBERT`-alapú modellel szemben. Viszont a `huBERT`-alapú + beam search modell jobb eredményt ért el, mint ahol nem volt alkalmazva a beam search, ahogy a 4. táblázatban is látható. A konklúzió tehát, hogy jobb a beam search, ezért a többi transformer-alapú modellnél is ezt használtuk alapértelmezett módszerként.

Így, mivel bevettük a vizsgálatba a régebbi rendszereket, ezért külön korpuszokra is le lettek mérve a modellek. Az `emBERT`-tel való összehasonlítás során a SzegedNER korpusz esetében szinte fej fej mellett van a két modell (`emBERT` és a beam search `huBERT`-alapú modell), de a NerKor esetében kicsit alulmarad a `huBERT`-alapú modell. A kombinált korpuszon is jobb az `emBERT`, mert a 12 réteg feletti Viterbi-algoritmus garantálja a jósolt címkeszekvencia konzisztenciáját. Ezen felül, ha megnézzük az összes transformer-alapú modellt és az `emBERT` eredményeit, kijelenthető, hogy az utóbbi továbbra is a leghatékonyabb.

4.5. `XLM-RoBERTa`-hu modell méretcsökkentése

Ahogyan a 3.3. fejezetben említettük, az `XLM-RoBERTa` modellek méretét, ha sikerülne csökkenteni megtartva a teljesítményét, akkor elérhető lenne egy rendkívül pontos modell jelentősen kevesebb memóriaigénnyel.

Az első fázisban reguláris kifejezések segítségével szűrtük ki azokat a tokeneket, amelyek nem fellelhetőek a magyar és az angol nyelvben. Így például a kínai szimbólumokat, török betűket, cirill betűket tartalmazó részszavakat és karaktereket kivettük. Ugyanakkor az Embedding layerből is eltávolítottuk a hozzátartozó vektorokat, így a paraméterszáma is csökkent a modellnek.

	Token	Mondatra bontás	PoS	Morf.	Lemma
`XLM-RoBERTa-base`	**99,89%**	99,33%	97,73%	96,61%	98,55%
`XLM-RoBERTa-base-hu`		98,56%	97,60%	96,27%	98,42%
`XLM-RoBERTa-large`	**99,89%**	**99,67%**	**98,01%**	**97,15%**	**98,67%**
`XLM-RoBERTa-large-hu`		97,24%	97,46%	95,84%	98,11%

5. táblázat. Módosított `XLM-RoBERTa` modellek F1-score eredményei az UD korpuszon. (Az aláhúzás az aktuális résztáblázatban a legjobb eredmény, a félkövérített pedig az adott egész táblázatban a legjobb eredmény.)

Ahogy a 5. és a 6. táblázatokban is észrevehető, a méretoptimalizált modellekkel hozzávetőlegesen sikerült tartani a teljesítményt. Jelenlegi kísérletek is hasznos eredményeknek számítanak, ugyanis az `XLM-RoBERTa-base-hu` esetén

	UAS	LAS	NER
`XLM-RoBERTa-base`	89,47%	86,02%	90,94%
`XLM-RoBERTa-base-hu`	89,21%	85,54%	89,89%
`XLM-RoBERTa-large`	**90,38%**	**86,88%**	**91,86%**
`XLM-RoBERTa-large-hu`	89,40%	85,46%	90,44%

6. táblázat. Módosított `XLM-RoBERTa` modellek UAS és LAS eredményei az UD korpuszon, a NER pedig a kombinált NER adatbázison lett mérve, amely F1-scoret mutatnak.

például sikerült a **felére csökkenteni** a méretet és az `XLM-RoBERTa-large-hu` modell pedig megközelítőleg **kétharmada az eredetinek**. Ezt a következő fejezetben jobban kifejtjük és összehasonlítjuk. Konklúzió, hogy a jelenlegi módszer segítségével képesek vagyunk csökkenteni a `XLM-RoBERTa` modell méretét minimális pontosságvesztéssel. Annak érdekében, hogy valóban hasznos legyen ez a modell és megtartsa a pontosságot, szükség lenne egy olyan heurisztikára, amely képes lenne ezt a feladatot ellátni. Jövőbeli tervünk, hogy az eldobandó szótöredékek halmazát statisztikai alapon válasszuk ki. Ehhez szeretnénk felhasználni a WebCorpus 2.0-át (Nemeskey, 2020b) és csak az abban előforduló leggyakoribb szótöredékeket megtartani, amivel a terveink szerint tovább csökkenthetnénk a szótár méretét és megtarthatnánk olyan speciális karaktereket is, amiket a kézi szabályokkal nem fedtünk le. A jelenlegi script elérhető publikus módon a GitHubon[4], ami lehetővé teszi egy új specifikus nyelvi modell készítését bármely nyelvre.

4.6. Memóriahasználat

A transformer architektúrák esetén fontosak az erőforrásigények, ezért ebben a fejezetben mélyebben megvizsgáljuk a futásidőket és a memóriaigényeket az összes eddigi modell szemszögéből. Az `UDPipe`, `emtsv` és az `UDify` rendszerek nem támogatják a névelem-felismerést, így ezeket a modelleket a névelem-felismerő nélkül mértük. A mérések az UD teszthalmazán készültek.

Először nézzünk meg a CPU-s oszlopot. A 7. táblázatban jól látható, hogy a transformer modellek a többi rendszerekhez képest több erőforrást igényelnek. Viszont kivételt képez például az `emtsv` gyorsasága a `huBERT`-alapú modellhez képest. A `Stanza` szintén lassabb, memóriaigények tekintetében pedig egy skálán mozognak. Az `UDPipe` ellenben mind a két esetben jól teljesített a futásidő és memóriaigények tekintetében, ugyanakkor ez a rendszer minden elemző esetén alulmarad hatékonyságban a többi rendszerhez képest. A többi transformer modell több paraméterrel rendelkezik, ezért és a többnyelvűség miatt is több erőforrást igényelnek.

Megfigyelhető az is, hogy a `huBERT`-alapú modell memóriahasználatban a `HuSpaCy-large` modelljéhez képest nem mutat jelentős eltérést. Ellenben az

[4] `https://github.com/huspacy/huspacy-resources/blob/master/scripts/XLM-RoBERTa_size_reduction/XLM-RoBERTa_size_reduction.ipynb`

	Gyorsaság (token/mp) CPU	Gyorsaság (token/mp) GPU	Memóriahasználat (GB)
`Stanza`	30	395	5,3
`UDPipe`	3175	–	1,3
`emtsv`	113	–	3,9
`UDify`	129	475	3,2
`HuSpaCy-large`	728	4685	3,2
`huBERT`	176	2605	4,8
`BERT-base-Multilingual`	138	2631	7,0
`XLM-RoBERTa-base`	151	2847	11,2
`XLM-RoBERTa-base-hu`	166	3265	6,2
`XLM-RoBERTa-large`	50	2353	18,9
`XLM-RoBERTa-large-hu`	59	2390	14,6

7. táblázat. A CPU-n (AMD EPYC 7F72) és GPU-n (NVIDIA A100 40 GB) összehasonlított rendszerek gyorsasága (token/másodpercben mérve) és a maximális memóriahasználatuk.

`XLM-RoBERTa` modellekre ez már nem teljesen igaz. Ahogyan szó volt a 3.1. fejezetben a modellek összetételéről, itt nyilvánul meg a három transformer modell mérete és az, hogy a többnyelvű modellek sokkal több memóriát igényelnek már önmagukban is.

A méretoptimalizált és az eredeti modellek között a futásidő azért nem változott drasztikusan, mert tulajdonképpen mind a két esetben ugyanazok a neuronok feleltek a futásidőért. Más szóval, a modell csak azon neuronokat használta fel a predikáláshoz, amelyekre a magyar nyelvhez szükség van. Az, hogy el lettek távolítva azok a részek, amelyeket a modell egyébként sem használt, nem befolyásolta a futásidőt.

5. Összegzés

Elkészítettünk a HuSpaCy-hez[5] egy `huBERT`-alapú modellt[6], ami minden hagyományos nyelvi elemzőt tartalmaz, vagyis tokenizálót, mondatra bontót, lemmatizálót, szófaji és morfológiai egyértelműsítőt, függőségi elemzőt, illetve ezeken felül névelem-felismerőt is és ezen eredmények teljes mértékben reprodukálhatóak.

Összességében a transformer-alapú modellek minden feladatban jobb eredményt érnek el a korábbi megoldásoknál. Ezek közül is a legtöbb feladatban a `XLM-RoBERTa-large` teljesített a legjobban, csupán a függőségi elemzés részfeladaton múlta felül a `huBERT`-alapú modell. Az alacsonyabb memóriaigényének és gyorsabb futásának köszönhetően a `huBERT` sok esetben jó választás lehet, de, ha a pontosságon van a hangsúly, a `XLM-RoBERTa-large` felhasználásával érhetjük el a legjobb eredményt. Viszont a state-of-the-art eredmények ellenére

[5] `https://github.com/huspacy/huspacy/tree/develop`
[6] `https://huggingface.co/huspacy/hu_core_news_trf`

is hátrányt jelenthet bizonyos végalkalmazásokban a nagyobb erőforrásigényük, például időérzékeny alkalmazásokban, illetve, ha korlátozott erőforrásokkal rendelkezünk. Ezekben a szituációkban megfelelő döntés lehet a `HuSpaCy-large` modell használata.

A függőségi elemzés és a névelem-felismerés problémákra több algoritmust is kipróbáltunk és a legmegfelelőbbet alkalmaztuk. Az eredmények alapján, a `huBERT`-alapú modell a függőségi elemző esetén hatalmas javulást mutat az elődeihez képest, így sikerült egy state-of-the-art eredményt elérni. Továbbá a morfológiai elemzés esetén is az `XLM-RoBERTa-large`-alapú modell képest volt 4 százalékponttal javulni az eddigi legjobb eredményhez képest.

Az elkészült modellek bárki számára szabadon igénybe vehetők a nyílt forráskódú HuSpaCy részeként.

Köszönetnyilvánítás

A kutatás az Európai Unió támogatásával valósult meg, az RRF-2.3.1-21-2022-00004 azonosítójú, Mesterséges Intelligencia Nemzeti Laboratórium projekt keretében. Továbbá szeretnénk megköszönni Berend Gábor segítségét a nyelvi modellek méretcsökkentésével kapcsolatban.

Hivatkozások

Berkecz, P., Orosz, G., Szántó, Z., Szabó, G., Farkas, R.: Hybrid lemmatization in HuSpaCy. In: XIX. Magyar Számítógépes Nyelvészeti Konferencia (2023)

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., Stoyanov, V.: Unsupervised cross-lingual representation learning at scale. arXiv preprint arXiv:1911.02116 (2019)

Csendes, D., Csirik, J., Gyimóthy, T.: The Szeged Corpus: A POS tagged and syntactically annotated Hungarian natural language corpus. In: International Conference on Text, Speech and Dialogue. pp. 41–47. Springer (2004)

De Marneffe, M.C., Manning, C.D., Nivre, J., Zeman, D.: Universal dependencies. Computational linguistics 47(2), 255–308 (2021)

Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)

Dozat, T., Manning, C.D.: Deep biaffine attention for neural dependency parsing. arXiv preprint arXiv:1611.01734 (2016)

Honnibal, M.: Introducing spaCy (Feb 2015), `https://explosion.ai/blog/introducing-spacy`

Indig, B., Sass, B., Simon, E., Mittelholcz, I., Vadász, N., Makrai, M.: One format to rule them all – The `emtsv` pipeline for Hungarian. In: Proceedings of the 13th Linguistic Annotation Workshop. pp. 155–165. Association for Computational Linguistics, Florence, Italy (aug 2019a), `https://www.aclweb.org/anthology/W19-4018`

Indig, B., Sass, B., Simon, E., Mittelholcz, I., Kundráth, P., Vadász, N.: `emtsv` — Egy formátum mind felett. In: Berend, G., Gosztolya, G., Vincze, V. (szerk.) XV. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY 2019). pp. 235–247. Szegedi Tudományegyetem Informatikai Tanszékcsoport, Szeged (2019b)

Kondratyuk, D., Straka, M.: 75 languages, 1 model: Parsing universal dependencies universally. arXiv preprint arXiv:1904.02099 (2019)

Medress, M.F., Cooper, F.S., Forgie, J.W., Green, C., Klatt, D.H., O'Malley, M.H., Neuburg, E.P., Newell, A., Reddy, D., Ritea, B., és mtsai: Speech understanding systems: Report of a steering committee. Artificial Intelligence 9(3), 307–316 (1977)

Nemeskey, D.M.: Egy emBERT próbáló feladat (2020a)

Nemeskey, D.M.: Introducing huBERT. XVII. Magyar Számítógépes Nyelvészeti Konferencia pp. 3–14 (2022)

Nemeskey, D.M.: Natural Language Processing Methods for Language Modeling. Ph.D.-értekezés, Eötvös Loránd University (2020b)

Nivre, J., Agić, Ž., Ahrenberg, L., Antonsen, L., Aranzabe, M.J., Asahara, M., Ateyah, L., Attia, M., Atutxa, A., Augustinus, L., és mtsai: Universal Dependencies 2.1 (2017)

Orosz, G., Novák, A.: PurePos 2.0: a hybrid tool for morphological disambiguation. In: Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013. pp. 539–545 (2013)

Orosz, G., Szántó, Z., Berkecz, P., Szabó, G., Farkas, R.: HuSpaCy: an industrial-strength Hungarian natural language processing toolkit. arXiv preprint arXiv:2201.01956 (2022)

Pires, T., Schlinger, E., Garrette, D.: How multilingual is multilingual BERT? arXiv preprint arXiv:1906.01502 (2019)

Qi, P., Zhang, Y., Zhang, Y., Bolton, J., Manning, C.D.: Stanza: A python natural language processing toolkit for many human languages. arXiv preprint arXiv:2003.07082 (2020)

Simon, E.: Approaches to Hungarian Named Entity Recognition. Ph.D.-értekezés, PhD School in Cognitive Sciences, Budapest University of Technology and Economics (2013)

Simon, E., Indig, B., Kalivoda, Á., Mittelholcz Iván, S.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)

Simon, E., Vadász, N.: Introducing NYTK-NerKor, A Gold Standard Hungarian Named Entity Annotated Corpus. In: Ekstein, K., Pártl, F., Konopík, M. (szerk.) Text, Speech, and Dialogue - 24th International Conference, TSD 2021, Olomouc, Czech Republic, September 6-9, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12848, pp. 222–234. Springer (2021)

Simon, E., Vadász, N., Lévai, D., Dávid, N., Orosz, G., Szántó, Z.: Az NYTK-NerKor több szempontú kiértékelése. XVIII. Magyar Számítógépes Nyelvészeti Konferencia (2022)

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 (Oct 2018), https://www.aclweb.org/anthology/K18-2020

Szarvas, G., Farkas, R., Felfoldi, L., Kocsor, A., Csirik, J.: A highly accurate Named Entity corpus for Hungarian. In: Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06) (2006a)

Szarvas, G., Farkas, R., Kocsor, A.: A multilingual named entity recognition system using boosting and C4.5 decision tree learning algorithms. In: International Conference on Discovery Science. pp. 267–278. Springer (2006b)

Váradi, T., Simon, E., Sass, B., Gerőcs, M., Mittelholtz, I., Novák, A., Indig, B., Prószéky, G., Vincze, V.: Az e-magyar digitális nyelvfeldolgozó rendszer (2017)

Váradi, T., Simon, E., Sass, B., Mittelholcz, I., Novák, A., Indig, B.: E-magyar–A Digital Language Processing System (2018)

Zsibrita, J., Vincze, V., Farkas, R.: magyarlanc: A Toolkit for Morphological and Dependency Parsing of Hungarian. In: Proceedings of Recent Advances in Natural Language Provessing 2013. pp. 763–771. Association for Computational Linguistics, Hissar, Bulgaria (2013)