

Szolár Balázs

BERT MODELLEK TELJESÍTMÉNYÉNEK VIZSGÁLATA BINÁRIS OSZTÁLYOZÁSI FELADATHOZ KAPCSOLÓDÓAN ANGOL ÉS MAGYAR NYELVŰ SZÖVEGKÖRNYEZETEBEN

Témavezető: Dr. Fogarassyné dr. Vathy Ágnes, Székér Szabolcs

Műszaki Informatikai Kar, Rendszer- és Számítástudományi Tanszék

Pannon Egyetem
Intézményi Tudományos Diákköri Konferencia
2021



Tartalomjegyzék

1.	Bevezetés.....	1
2.	Elméleti háttér	2
2.1.	Természetes nyelvfeldolgozás.....	2
2.2.	Figyelem alapú modellek	5
2.3.	Transformer	6
2.4.	BERT.....	8
2.4.1.	Előtanítás	9
2.4.2.	Finomhangolás	10
2.4.3.	Előtanított BERT modellek	12
3.	Kutatási célkitűzés és módszertan	15
3.1.	A szöveghalmaz és a kapcsolódó feltáró elemzés.....	15
3.2.	Kutatási módszertan	19
3.2.1.	A szöveghalmazok előkészítése	19
3.2.2.	Alkalmazott modellek	22
3.2.3.	A modellek paraméterhangolása	23
3.2.4.	Tesztscenáriók	24
3.2.5.	Kiértékelés módszertana.....	25
3.2.6.	A kutatás technikai háttere	27
4.	Eredmények.....	28
4.1.	A BERT modellek hiperparaméter-hangolása és eredményei	28
4.2.	Tesztelési scenáriók eredményei	32
4.2.1.	Szenárió I: Modellek bilingualitása	33
4.2.2.	Szenárió II: Kiegyensúlyozás hatása	36
4.2.3.	Szenárió III: A szöveghalmaz méretének hatása az osztályozás teljesítményére	39
4.2.4.	Szenárió IV: Cased, Uncased és Lowercased esetek vizsgálata.....	41
5.	Összefoglalás.....	44
	Irodalomjegyzék	46
	Mellékletek	49

1. Bevezetés

Az utóbbi években a Bidirectional Encoder Representations from Transformers, azaz BERT modell kiemelkedő eredményeket ért el különféle természetes nyelvfeldolgozási feladatokban, mint például a kérdésmegválaszolás vagy osztályozás. Mindez részben a transfer learning metodikának köszönhető. A transfer learning lényege, hogy egy ismert feladaton (mint például gépi látás esetén, az ImageNet) előtanítanak egy neurális hálózat alapú modellt, majd egy új feladaton finomítják és alkalmazzák azt. Mostanra ezen módszer alkalmazása azonban a gépi látás mellett a természetes nyelvfeldolgozás területén is elterjedt.

Jelen TDK keretei között célom megvizsgálni 2 különböző nyelvre (angol és magyar) előtanított BERT modell hatékonyságát, egy új, 2 osztályos, osztályozási feladaton, majd ennek eredményeit összehasonlítani és kiértékelni. Ebből a célból egy új adathalmazt hozok létre, melyhez a Steam nevű online videójáték áruházból gyűjtök értékeléseket. Ezen a portálon a felhasználók szöveges értékelést írhatnak egy játék megvásárlása után, és ehhez egy „Ajánlott” vagy „Nem ajánlott” címkét kötnek. Ezt az adathalmazt különböző előfeldolgozásokkal vizsgálom meg, és kiértékelem, hogy ezek milyen hatással vannak a modellek eredményére, úgy, hogy hiperparaméter hangolással kiválasztom a legjobb beállításokat.

A munka felépítése a következő: először az Elméleti háttér fejezetben bemutatom a természetes nyelvfeldolgozás témakörét és a szóbeágyazást általánosságban, majd a BERT elméleti működését és a hozzá kapcsolható megelőző modelleket, valamint az eddigi gyakorlati alkalmazásait. A Kutatási módszertan bemutatása fejezetnél részletesebben írok az adathalmazról, a kísérletekhez használt BERT modellekről és magáról a kísérletek elvégzésének módszeréről. Az Eredmények és az Összefoglalás részben pedig bemutatom és összefoglalom a kísérletek eredményeit és következtetéseit.

2. Elméleti háttér

2.1. Természetes nyelvfeldolgozás

A természetes szöveg feldolgozása más folyamatot igényel, mint például a programozási nyelvek értelmezése. Jelentős különbség, hogy a természetes nyelveknek rugalmas szerkezete és felépítése van, melyhez tudnia kell alkalmazkodni az azokat feldolgozó algoritmusoknak.

A természetes nyelvfeldolgozáson (Natural Language Processing, NLP) alapuló rendszerekben általában nem karakter szintű szövegkezelés zajlik, hanem a szavakat veszik a legelemibb egységeknek. Ez azt eredményezi, hogy nem kell foglalkoznunk a szavak karakterek szerinti felépítésével, hiszen ezek nem mindig hordoznak jelentésbeli információt (pl. kutya – eb, azonos jelentésű, mégis más hangalakú szavak).

Az NLP-ben alapvetően 2-fajta megközelítési módszer használatos.

- A **számolás alapú metódusok** megszámlálják, hányszor szerepel egy-egy szó a szomszédos szavakkal együtt egy nagy szöveggyűjteményben, majd ezeket a statisztikákat egy kis, sűrű vektorba ágyazzák be.
- A **prediktív metódusok** viszont a szavakat közvetlenül próbálják megjósolni a szomszédos szavakból. Ezt megtanult, beágyazott vektorok alapján teszik, melyek a modell paraméterei lesznek. Így az ilyen metódusoknak a célja ezeknek a paramétereknek a beállítása úgy, hogy a modell a megfelelő szavakat jósolja meg.

A fenti módszerek vizsgálata széleskörű kutatások tárgyát képezi. Baroni és társai [1] megállapították, hogy a prediktív metódusok jobban teljesítenek a számolás alapúaknál.

Ahhoz, hogy a szavakat beágyazott vektorokkal reprezentáljuk, szóbeágyazást (word embedding) [2] kell használnunk. E módszer lényege, hogy a szavakat egy n dimenziós vektortérbe ágyazzuk be úgy, hogy a hasonló jelentésű szavak közel kerüljenek egymáshoz. Annak eldöntését pedig, hogy mely szavak hasonló jelentésűek, a disztribúciós hipotézisre bízunk. Ez a hipotézis arról szól, hogy

A módszer másik érdekes tulajdonsága, hogy miután a szóbeágyazás segítségével megkaptuk a beágyazott vektorokat, alkalmazhatunk rájuk műveleteket is, és ezáltal az ember számára is intuitív megoldásokat kaphatunk. Tekintsük például a következő műveletet:

$$\textit{király} - \textit{férfi} + \textit{nő} \quad (1)$$

Ez egy olyan vektort eredményezhet, mely a vektortérben közel áll a „*királynő*” szó beágyazott vektorához.

A fő kérdés tehát az, hogy mennyire jó szóbeágyazásokat tudunk készíteni, ettől fog függeni, hogy mennyire jól teljesítünk olyan feladatokban, ahol ezeket a beágyazásokat használjuk fel (ilyen lehet például mondatok osztályozása, mondatok hasonlóságának meghatározása, kérdés megválaszolás stb.).

A szóbeágyazás témakörében, egy nagy áttörést elérő módszer volt korábban a *word2vec* [4], mely a prediktív metódusok csoportjába tartozik. A *word2vec* kétfajta módszert alkalmaz, melyek a *folytonos szósák modell* (Continuous Bag-of-Words, CBOW) és a *skip-gram modell*. A két modell abban különbözik egymástól, hogy a CBOW modell a cél szavakat (pl. alma) próbálja megjósolni, a kontextus szavakból (pl. „*leesett a fáról egy ____*”), a skip-gram modell pedig ezt fordítva teszi, tehát a cél szóból próbálja megjósolni annak kontextusát. Ennek a két feladatnak az elvégzésére mindkét modell egy 2-rétegű neurális hálót használ.

Az ehhez hasonló módszerekkel (pl.: GloVe [5], Fasttext [6]) készült szóbeágyazások komplexitását az adja, hogy nem csak szavak megkülönböztetése a feladatuk, hanem az is, hogy szavak sorozatának a jelentését megértsék, hiszen csak így érthetik meg az adott nyelvet. Ezért nagy mennyiségű adathalmazra van szükség ahhoz, hogy ezekkel a modellekkel jó eredményeket érjünk el.

Az előbb említett statikus módszerek továbbfejlesztései a dinamikus modellek, melyek fő jellemzője, hogy használat előtt egy általánosabb halmazon vannak betanítva. Ezek a modellek hatalmas adathalmazokon végeznek el feladatokat (pl. Stanford Question Answering Dataset [7]), így magasabb szintű értelmezést tanulnak meg, majd ezeket a betanított modelleket később újrahasználik egy feladat-specifikus adathalmazon. Ezt a módszert *transfer learning*-nek nevezzük,

mivel az általános tudást, amit a nagy adathalmazon megtanult a modell, átvisszük a specifikus feladatra.

Az egyik legjobban teljesítő ilyen dinamikus modell a Google által készített, nyílt forráskódú BERT [8], viszont megértéséhez először érdemes a *figyelmi modelleket* és a *Transformert* is részletesen áttekinteni.

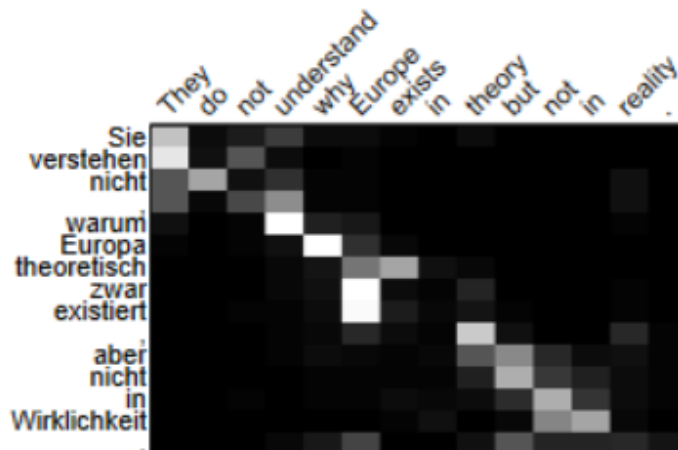
2.2. Figyelem alapú modellek

A *figyelmi mechanizmusok* nem neurális modellek, sokkal inkább „bővítmények” a már meglévő modellekhez. Ahogy a nevük is mutatja, az emberi figyelem ihlette őket, tehát azt a képességet adják a neurális modelleknek, hogy a bemenetük egyes régiójára (esetleg több régiójára) jobban fókuszáljanak.

A figyelem mechanizmus Bahdanau és társai által született meg 2015-ben [9], hogy megoldja a *seq2seq* modellek kritikus problémáját, vagyis azt, hogy képtelenek hosszú mondatokat megjegyezni. A *seq2seq* megközelítést Sutskever és társai publikálták 2014-ben [10], s az ezen elven alapuló modellek elsősorban fordításra használhatók. A *seq2seq* modellek azonban sajnos gyakran elfelejtik a mondat első részét, mire végeznek a teljes bemenet feldolgozásával. A modell neve onnan származik, hogy szekvencia bemenetből szekvencia kimenetet generál. Általában egy enkóder-dekóder architektúrából áll, ahol az enkóder rész feladata, hogy a bemenetből egy fix hosszúságú „gondolat” (vagy kontextus) vektort generáljon, mely lehetőleg elég jól összegzi a bemenet jelentését. A dekóder pedig ezt a kontextus vektort használja fel, hogy az átalakított kimenetet létrehozza, fordítás esetén a másik nyelven. Mind az enkóder, mind a dekóder rész rekurrens neurális hálóból (RNN) állnak általában, így a kontextus vektor az utolsó enkóder RNN rejtett állapota lesz.

A fordítási feladatokhoz használt *seq2seq* modell annyiban változik egy figyelem mechanizmus alkalmazásával, hogy az enkóder utolsó rejtett állapotának felhasználása helyett, az összes rejtett állapotot felhasználjuk a kontextus vektor létrehozásához. Ezek az állapotok egy-egy súlyt kapnak, ennek segítségével fogja tudni a modell megadni az egyes bemenetek fontosságát. A súlyoknak az értéke pedig egy olyan egyszerű előrecsatolt neurális háló segítségével adható meg, mely

egy rejtett réteget tartalmaz. A modell tanulása után ezeket a súlyértékeket vizualizálhatjuk, ami egy betekintést nyújt számunkra arra vonatkozóan, hogy az egyes lépésekben éppen mire figyel a modell (2. ábra).



2. ábra Egy angolról-németre fordító modell súlyértékei tanulás után (a fehér a magas érték) [6]

A figyelmi mechanizmusoknak több variációjuk is van, mely alapján *lágú* (soft) vagy *kemény* (hard) figyelmi mechanizmusokról beszélhetünk. A lágú és a kemény figyelem közötti különbség az, hogy míg a lágú figyelemben minden rejtett állapotot felhasználunk a kontextus vektor elkészítéséhez csak más mértékben, addig a kemény változatnál csak az egyik rejtett állapotot választjuk ki, a többi pedig eldobjuk. A kemény figyelem hátránya, hogy nehezebben differenciálható, viszont kevesebb számítást igényel.

2.3. Transformer

A *Transformer* egy Google által fejlesztett modell [11], az előbb leírt seq2seq figyelmi modell továbbfejlesztése. Két fő erénye, hogy jobb eredményeket ér el fordítási feladatokban és jobban párhuzamosítható, kevesebb tanítási időt igényelve.

Hasonlóan a seq2seqhez, itt is enkóderből és dekóderből áll az architektúra, viszont itt már az enkóder komponens 6 db egymásra helyezett enkóderből áll, a dekóder pedig ugyanennyi dekóderből.

Az enkóderek sem egyszerű RNN-ek már, hanem egy *ön-figyelmi* (Self-Attention) és egy *előrecsatolt neurális háló* (Feed Forward Neural Network, FFNN)

részből állnak. Az ön-figyelmi rész segít eldönteni, hogy egy adott szó esetén, melyik másik szavakkal szeretnénk foglalkozni a bemeneti szekvenciából. Ennek a kimenetét kapja meg az FFNN, viszont az ön-figyelmi résszel ellentétben a bemenetek itt már függetlenek egymástól.

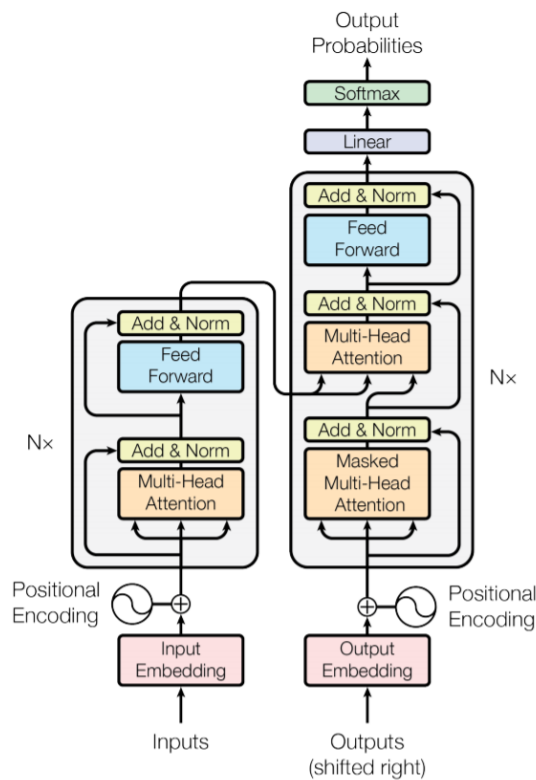
Az FFNN koncepciója egyértelmű, viszont az ön-figyelmi komponens magyarázatot igényel. A bemenete egy egész szekvencia (az első komponensnél ez maga a mondat), és a feladata, hogy a szekvencia minden eleméhez hozzátartozó a szekvencia többi elemét is valamilyen súllyal. A figyelem analógiájával, a nagyobb súlyú elemekre (szavakra) fog nagyobb figyelmet fordítani a modell.

A gyakorlatban ehhez a feladathoz 3 mátrix szükséges, melyeket a modellel együtt tanítunk. Egy-egy vektort (szót) ezekkel szorzunk be, így 3 újabb vektort kapunk. Mindegyiknek speciális feladata van, így külön elnevezésük is van: *Query*, *Key* és *Value*. A figyelmi réteg számolása a következő: legyen $\mathbf{X}$ egy mátrix, melynek sorai az adott bemenet elemei. Legyen $\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V$ az említett 3 súlymátrix. Ekkor, $\mathbf{X}\mathbf{W}^Q = \mathbf{Q}$, $\mathbf{X}\mathbf{W}^K = \mathbf{K}$, $\mathbf{X}\mathbf{W}^V = \mathbf{V}$ a Query, Key és Value értékeket tartalmazó mátrixok lesznek, d_k pedig legyen ezen oszlopainak a száma. Ezekkel kiszámítható a figyelmi komponens kimenete:

$$Attention(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = softmax\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V} \quad (2)$$

Ezt az eredeti cikkben [11] kibővítették „többfejű” -figyelemre (multiheaded attention) is, mely nem csak 3 súlymátrixot használ, hanem N -szer ennyit, ahol N a „fejek” száma, és minden bemenetre, ennyiszer számolunk kimenetet külön-külön minden hármásra. Az eredmény N db kimenet, amit emiatt összegezni kell. Ezt a kimenetek összefűzésével, majd egy újabb $\mathbf{W}^0$ súlymátrixal való beszorzással tehetjük meg. Az alkotók cikkükben [11] 8 db párhuzamos figyelmi réteget (fejet) használtak.

Mivel a modell nem használ ismétlődést vagy konvolúciót, ezért a szavak sorrendjét külön kell bevinni a modellbe. Ezt a cikkben egyszerű szinusz, koszinusz függvényekkel létrehozott vektorokkal teszik, ezeket adják hozzá a bemenethez.



3. ábra A Transformer részei – a bal szürke alakzat az enkóder a jobb pedig a dekóder [11]

Ahogy a 3. ábra mutatja, az enkódereknek még egy segédkomponense van, ez az összeadás és normalizálás (Add & Norm), mely az előző komponens bemenetét összeadja a kimenetével, majd azt normalizálja.

Miután az enkóderek befejezték működésüket, a dekóderek egyrészt megkapják az előző lépések *Key* és *Value* mátrixait, másrészt pedig az eddigi kimeneteket. A dekóder rész feladata lesz megjósolni a kimenet következő szavát, az előző kimenetek alapján. Emiatt van végül egy egyszerű neurális háló a szótárunk méretével egyenlő kimenetszámmal és egy softmax függvény, hogy a kimenetünkben valószínűségek legyenek. Így minden szóra megkapjuk, hogy mekkora valószínűséggel fog következő elemként szerepelni.

2.4. BERT

A Transformernek készült transfer learningre specializált verziója is, *OpenAI Transformer* néven, mely a Transformernek csak a dekódoló részét használja fel

[12]. Részben erre, és más transfer learninges modellekre épít a BERT [8], melyet Devlin és társai publikáltak 2018-ban.

Jacob Devlin és társai 2 modell architektúrát mutatnak be cikkükben: *BERT-Base* és *BERT-Large* (később többet is közzétettek, melyeket később említek). Mindkettő a Transformer enkódereit használja csak, hiszen a cél csak egy nyelvi modell generálása. Viszont az első architektúra ebből 12-öt, a második pedig 24-et használ. Ezeknek az enkódereknek ugyanúgy vannak előrecsatolt neurális hálói és figyelmi rétegei (fejei), ezek száma szintén különbözik a 2 architektúrában.

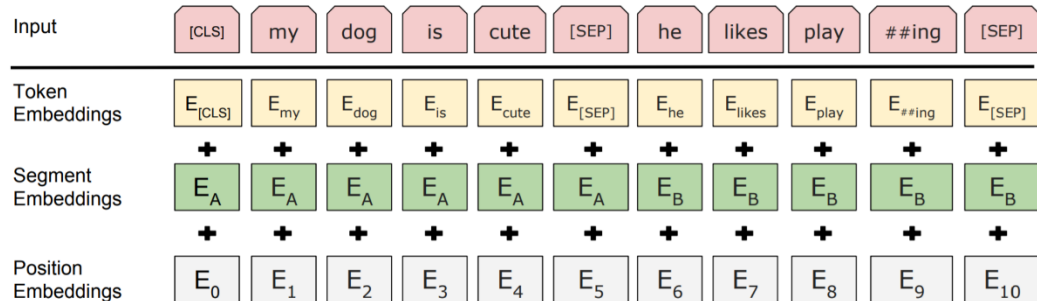
2.4.1. *Előtanítás*

Ahogy korábban említettem, a transfer learninges modellek használatának 2 fázisa van. Először az előtanítás (pre-training) során egy hatalmas adathalmazon különböző feladatokat tanítunk be a modellnek, hogy legyen egy általános megértése a nyelvről. Majd ezután használhatjuk az előre betanított modellt ún. „downstream” feladatokra. Ez a finomítás (fine-tuning) fázis, ami már sokkal gyorsabb és kevesebb adatot igényel.

A BERT-nél az előtanítás során 2 feladatra tanítják be a modellt. Az első a szó predikció, mely a nyelvi modell megtanulására szolgál, ezért hívják **maszkolt nyelvi modellnek** (Masked Language Model). A seq2seq és a Transformer eddig mind csak egyirányúak voltak, tehát csak balról-jobbra vagy fordítva tudták előrejelezni a következő szót. A Transformer esetében ez a korlátozás a dekóder esetében élt, viszont a BERT-nél nincs használva dekóder. Így a BERT-nél nem létezne ez az akadály, viszont egy másik probléma a kétirányúsággal, hogy ha egyszerre látja a modell az összes szót, akkor nem lehet a szópredikciót értelmezni (mivel mindent lát, így bármelyik szó predikciója triviális lenne). Erre a BERT megoldása a bemenet maszkolása, innen jön a feladat nevének „maszkolt” része. A bemenet véletlenszerű 15% százalékát a [MASK] tokennel helyettesítik (ez a feladattípus nem új, másik munkában Cloze feladatnak hívták [13]). A feladat tehát ezeknek a maszkolt szavaknak a predikciója.

A második előtanító feladat a **következő mondat jóslása** (Next Sentence Prediction, NSP), mivel sok downstream feladathoz szükség van a mondatok

közötti kapcsolat megértésére. Ehhez át kell alakítani a bemenetet, amit a 4. ábra mutat.



4. ábra A BERT bemenetének feldolgozása NSP-hez [8]

Ahogy a 4. ábra mutatja, 2 mondat van összefűzve, az elhatárolásukra pedig a [SEP] token szolgál. Szegmens beágyazásokat adunk hozzájuk, melyek megmondják, hogy az adott szó melyik szegmenshez tartozik. Végül pedig, a Transformaterekhez hasonlóan, itt is hozzájuk adunk vektorokat, melyek a szavak sorrendjéről tartalmaznak információt.

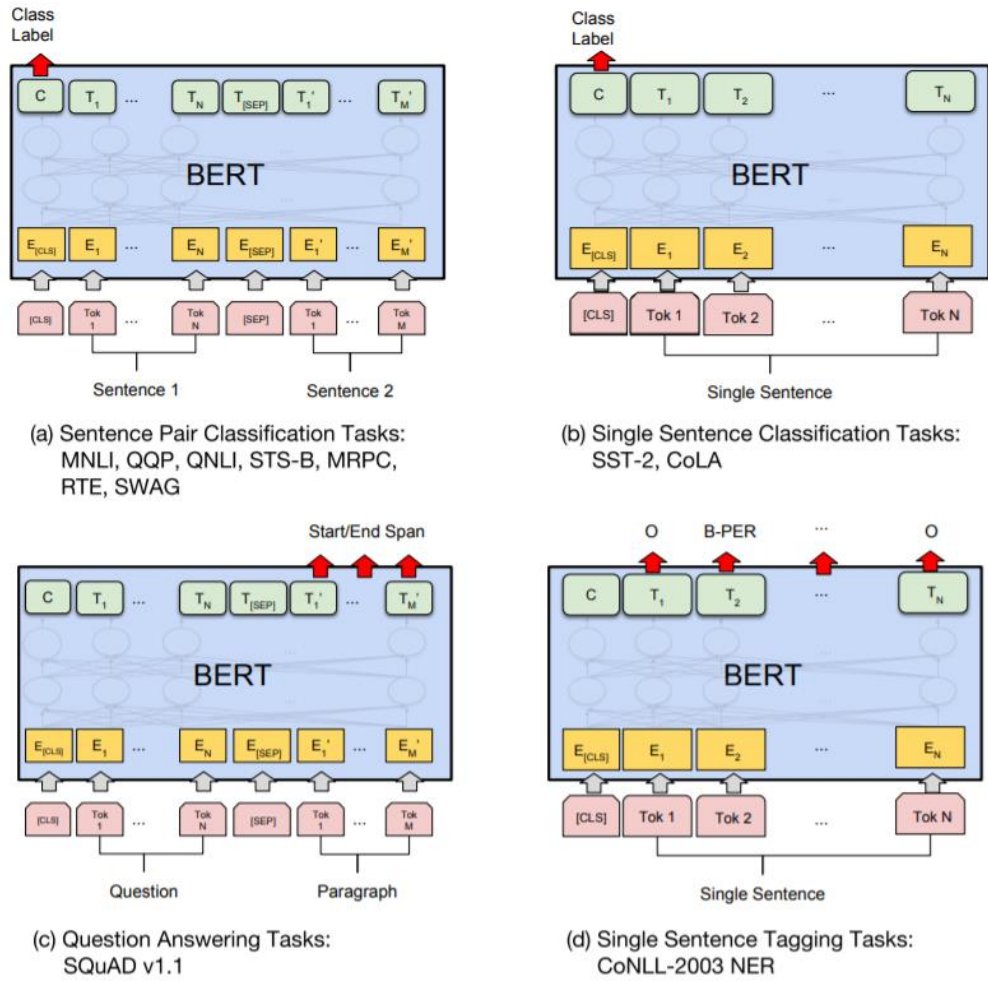
A bemenet elejére a [CLS] tokent (CLS, mint classification) is hozzáadjuk és mivel a következő mondat jóslás feladat célja osztályozás, ezért a kimenetben csak ezt vesszük figyelembe. Egy FFNN, majd softmax rétegnek átadva megkapjuk annak a valószínűségét, hogy a 2 bemeneti mondat egymást követi.

A BERT modellben a két feladat megtanítása egyszerre történik, tehát az előbb leírt feldolgozáshoz még a maszkolást is elvégezzük, a kimenetben pedig csak a [CLS] és a maszkolt tokeneket használjuk fel.

2.4.2. Finomhangolás

A finomhangolás (fine tuning) során az előtanítás részben elkészült modell felhasználásával egy specifikusabb feladaton lehet betanítani a modellt. Ilyen előtanított modelleket a BERT cikk [8] részeként is közzétettek több méretben és nyelvre vonatkozóan is. Az angol nyelven kívül például magyar nyelven is készült előtanított modell [14] 2020-ban (huBERT).

A BERT modell kis mértékű átalakításával többféle downstream feladatot is el lehet végezni, erre az 5. ábra mutat példákat.



5. ábra BERT használata lehetséges downstream feladatokhoz [8]

Az 5. ábra (a) és (b) részén osztályozásra láthatunk példát. Ezen esetekben csupán a bemenetet kell változtatni és a kimenetben csak a [CLS] token kell figyelembe venni. A (c) modell kérdés megválaszolásra használható. Az eredeti cikkben [8] erre a stanfordi SQuAD adathalmazt (Stanford Question Answering Dataset) használták. A (d) modell pedig példa named-entity recognition (NER) felhasználásra.

Ezen túl a BERT-ből kizsedhetőek az egyes szavak beágyazásai is, így szóbeágyazóként is használható, mint például Han Xiao „Hungarian layer: A novel interpretable neural layer for paraphrase identification” című cikkében [15], ahol BERT csak egy nagyobb architektúra része. Egyszerűen minden egyes enkódoló

réteg kimenetét kigyűjthetjük, amin az adott szó keresztülment. Persze kérdés, hogy ezek közül melyiket érdemes használni. A cikk 6 opciót vizsgált meg, és a NER feladaton az utolsó 4 réteg kimenetének összefűzése adta a legjobb eredményt.

2.4.3. Előtanított BERT modellek

A transfer learning-es modellek esetében általában nem szükséges az előtanítás elvégzése, mivel ez adat-, idő- és erőforrásigényes, és mivel szabadon elérhetőek különböző adathalmazokon előtanított modellek.

Ahogy már említettem az előző fejezetben, Jacob Devlin és társai több előtanított modellt is megjelentettek a cikkük megjelenése óta. Adathalmaz szempontjából 3 típus készült, angol, kínai és többnyelvű. A többnyelvű modellhez a 100 legnagyobb Wikipédia tartalommal rendelkező nyelvet választották, és ezek Wikipédia cikkeit használták fel, ebbe a listába a magyar nyelv is beletartozik. Az angol modell szintén Wikipédia cikkekkel lett előtanítva, egy könyveket tartalmazó adathalmaz mellett. Egy másik paramétere ezeknek a modelleknek a méretük. Az eredeti cikkben csak Base és Large modellek szerepeltek, viszont azóta 24 különböző méretűt is megjelentettek, mint a Tiny, Mini, Small és a Medium, ezek réteg- és paraméterszámban különböznek. Ezeknek a kisebb modelleknek az előnye, hogy kevésbé erőforrásigényesek, így olyan intézmények, kutatók is használhatják a BERT-et, akik kevesebb számítási kapacitással rendelkeznek. Hátrányuk viszont, hogy a kisebb modellmérettel rosszabb eredményeket lehet csak elérni, például a General Language Understanding Evaluation (GLUE) [16] benchmarkon a Tiny modell csak 64,2 pontot ért el, a Large viszont 80,5-öt. Az utolsó paramétere ezeknek a modelleknek az előfeldolgozásukhoz kapcsolódik. Az *uncased* előtagú modellek előfeldolgozásánál kisbetűssé alakították a bemenetet és lecserélték a diakritikus jeleket, a *cased* előtagú modellekénél viszont nem. A finomítás adathalmazától függ, hogy van-e a kisbetű/nagybetűnek jelentősége szintaktikailag vagy szemantikailag, és ez alapján érdemes választani.

A fent említett modellek elég általánosak ahhoz, hogy a legtöbb downstream feladatnak megfeleljenek, de specifikus feladatokon jobb eredményeket lehet elérni, ha az előtanító adathalmaz is specifikus. Például, amennyiben rendelkezésre áll egy nagy méretű egészségügyi adathalmaz, ezt érdemes használni egy például

orvosbiológiai problémához. Ugyanígy, egy magyar nyelvű downstream feladathoz érdemesebb egy magyarra specializálódott modellt használni a többnyelvűvel szemben. Ilyen a huBERT is [14], mely a magyar Wikipédián és a Hungarian Webcorpus 2.0-án [14] volt előtanítva. Itt szintén elérhető egy cased model, viszont az uncased-et egy lowercased modell pótolja, mivel a magyar nyelvben a diakritikus jelek megkülönböztetőek. A huBERT természetes előnye, hogy magyar nyelven előtanított, így várhatóan jobban teljesít a finomítás alatt magyar adathalmazokon.

A BERT működése elég általános ahhoz, hogy sokféle downstream feladathoz alkalmazható legyen (ahogy az 5. ábra is mutatja), ezért sokrétű alkalmazási módjai léteznek.

Egy ilyen, az exBAKE [17], egy automata álhír detektáló modell, mely cikkek címe és tartalma közötti kapcsolatot elemez. A szerzők saját előtanított modellt állítottak elő a BERT-Base újságcikkekkel való kibővítésével, majd egy álhír adathalmazon finomították, így javítva az eddigi legkorszerűbb modellek 0,14-es F1-mérték értékén.

Egy másik esetben klinikai területen névelem-felismerésre használták [18]. Itt is új előtanított modellt készítettek 1,5 millió címkézetlen elektronikus egészségügyi nyilvántartásból, melyet EhrBERT-nek neveztek el. Ez a modell 3 orvosbiológiai, adathalmazon finomítva túlteljesítette az alap BERT-et (az alap BERT előbbi alfejezetben említett általánossága miatt) és az orvosbiológiai adathalmazon tanított BioBERT [26] F1 értékét is.

A BERT4Rec nevű modell [19] példa a BERT egy ajánlórendszerként való használatára. Ez eltérő a BERT alapvető NLP-s feladataitól, viszont a modell mégis megfelel ehhez a problémához is, hiszen itt is szekvenciákat tekintünk bemenetnek, kimenetnek viszont a szavak sorozata helyett, egy vásárló vásárlási tevékenységének a sorozatát. Ez a modell szintén túlteljesíti több adathalmazon is az eddigi szekvenciális ajánlómodelleket.

Ahogy már említettem, Han Xiao „Hungarian layer: A novel interpretable neural layer for paraphrase identification” című cikkében [15] szóbeágyazóként van használva a BERT. A magyar módszerrel kibővítve olyan, a parafrázisazonosítás

problémáját megoldó architektúrát hoz létre, mely jobb eredményeket ér el, mint az eddigi modellek. Például a Quora kérdéspárok adathalmazán a BERT-Base 70,1-es F1 értéket ért el, viszont a Hungarian layerrel kibővített modell 71,2-t. Ebben a példában is látható, hogy a BERT átalakítva és más modellbe beépítve is tud új eredményeket adni.

3. Kutatási célkitűzés és módszertan

Munkám célja a magyar és angol nyelven előtanított BERT modellek teljesítményének kiértékelése és összehasonlítása magyar és angol nyelvű szövegek bináris osztályozása során. Ehhez olyan nagyméretű, saját gyűjtésű szöveghalmazt hoztam létre, amely mind magyar, mind angol nyelvű szövegrekordokat tartalmaz, s az egyes szövegrekordok bináris osztálycímkével vannak ellátva.

Kutatásom során a főbb kutatási kérdések, amelyekre a választ kerestem, a következők voltak:

- Milyen teljesítményt nyújtanak az angol és magyar nyelven előtanított BERT modellek egymáshoz képest, ha azonos problémakörben a saját nyelvükön alkalmazzuk őket?
- Mennyire hatékony a BERT, ha a finomhangolása más nyelven történik, mint az előtanítása?
- Milyen hatással van a tanító és kiértékelő adathalmaz kiegyensúlyozatlansága a modell eredményére?
- Milyen hatással van az adathalmaz mérete a BERT modell eredményére?
- Mi a különbség a cased és az uncased/lowercased BERT modellek hatékonysága között?

Ebben a fejezetben a kutatás részére előállított szöveghalmaz, annak különböző módon előfeldolgozott verziói, az ezeket felhasználó BERT modellek, és a kutatási eredmények kiértékelésének módszertana kerül bemutatásra.

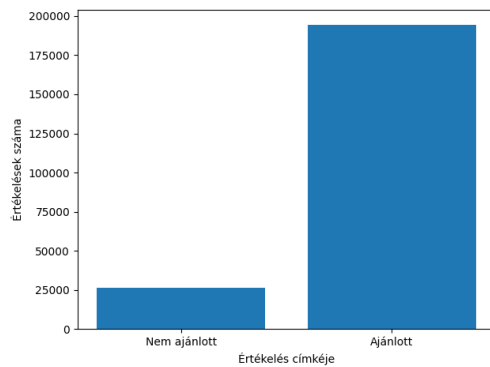
3.1. A szöveghalmaz és a kapcsolódó feltáró elemzés

A kutatáshoz létrehozott szöveghalmaz a Steam nevű online videójáték webáruházból [20] gyűjtött értékelésekből áll. A Steam alapvetően egy tartalomtovábbító és -kezelő rendszer, mely számítógépes szoftverek, nagyrészt videójátékok online eladásával és disztribúciójával foglalkozik. Mindegyik szoftvernek saját oldala van, ahol részletek találhatóak a termékről, és amennyiben egy felhasználó már megvásárolta azt, lehetősége van szöveges értékelést is írni

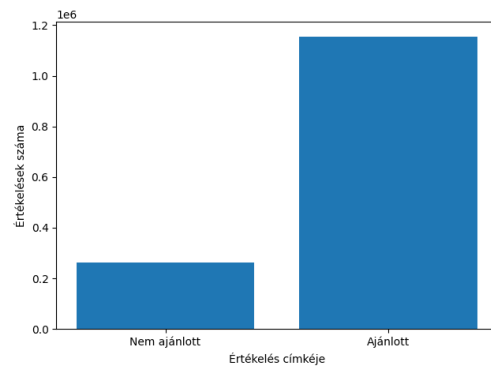
róla. Ehhez beállítható, hogy milyen az értékelés nyelve, és hogy a játékot az „Ajánlott” vagy „Nem ajánlott” kategóriába sorolja-e. Ezt az értékelést később más felhasználók szintén értékelhetik, „Hasznos” vagy „Nem hasznos”-ként, és kommenteket is írhatnak hozzá amennyiben ezt a véleményíró engedélyezte. Ebben a kutatásban a jellemzők közül csak a játék értékeléséhez megadott szöveget (*szöveg*), az értékelés nyelvét (*nyelv*), és az ajánlascímkét (*ajánlascímke*) használtam fel, mivel ezek kötelezően kitöltendő mezők. A játék értékelésénél megadott „Ajánlott”/„Nem ajánlott” címke az osztályozási feladatban bináris osztálycímkeként került felhasználásra.

A szövegek legyűjtése egy API révén valósult meg [21]. Az API alkalmazásával a Steamen 2021.03.04.-ig elérhető összes szoftverhez 2 lépésben letöltöttem az értékeléseket. Az értékeléskor beállított nyelv alapján először a magyar, majd az angol értékeléseket töltöttem le. Amíg a magyar adathalmaz nem nagy méretű és sikeresen letöltöttem az összes elérhető értékelést, addig az angol értékelések gyűjtését korábban leállítottam, mivel így is elegendő adat született. A legyűjtött magyar adathalmaz 221 232 értékelést, az angol pedig 1 418 632 értékelést tartalmaz. A továbbiakban ezen 2 adathalmaz tulajdonságainak bemutatása következik.

A kutatás alapját egy osztályozási feladat megvalósítása adja, ahol az ajánlascímke értékének prediktálása a cél. Ezért első lépésben érdemes megnézni, hogy milyen arányban jelennek meg az „Ajánlott” és a „Nem ajánlott” értékek az adathalmazban. Ezt mutatja be a 6. és 7. ábra.



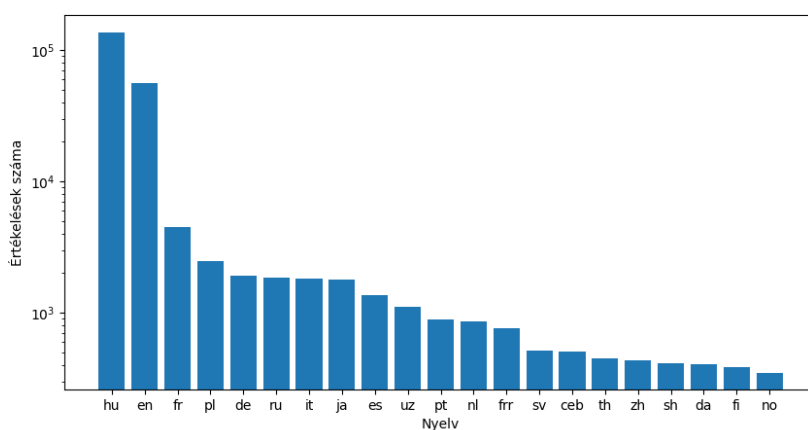
6. ábra Magyar adathalmaz értékeléseinek száma címke szerint



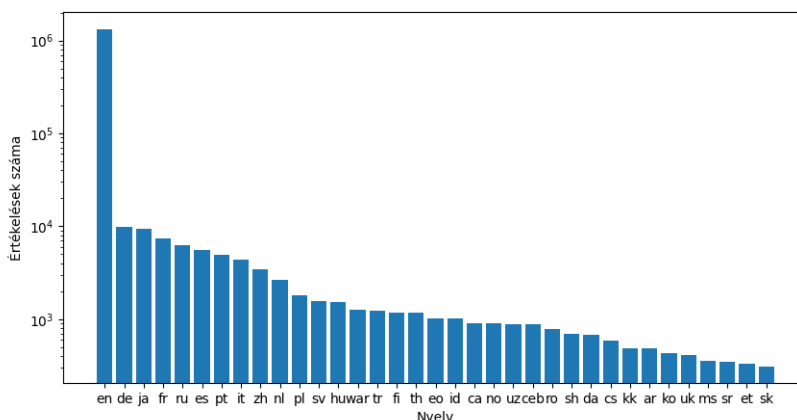
7. ábra Angol adathalmaz értékeléseinek száma címke szerint

Látható, mindkét adathalmaz kiegyensúlyozatlan, túlnyomó többségben vannak a magyar és az angol nyelvű értékelések esetében is a pozitív („Ajánlott”) értékelések.

A szöveges értékelés nyelvét meghatározó *nyelv* mező értékét a felhasználó szabadon választja ki egy legördülő listából, így előfordulhat, hogy nem ugyanazon a nyelven írja meg az értékelését, mint amit a listából kiválasztott. Ennek ellenőrzésére a fastText szövegosztályozás és reprezentáláshoz készített könyvtár nyelvdetektáló modelljét alkalmaztam [22, 23]. A segítségével kinyertem az értékelések valós nyelveit, melyek gyakorisága a 8. és 9. ábrán látható (az y tengely logaritmikus).



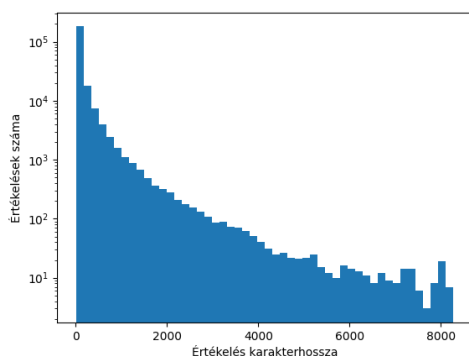
8. ábra Magyar adathalmaz nyelvi eloszlása a valós nyelv szerint



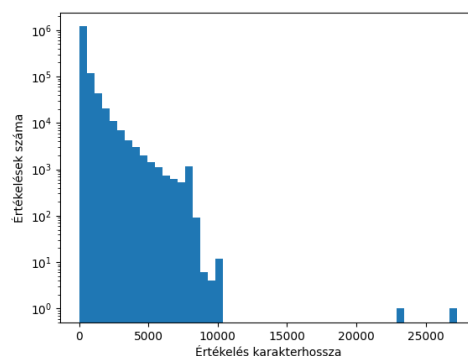
9. ábra Angol adathalmaz nyelvi eloszlása a valós nyelv szerint

Az ábrákon, az átláthatóság végett, ki lettek szűrve azok a nyelvek, amikhez 300-nál kevesebb értékelés tartozik. Az angol szöveghalmazban a bejegyzések túlnyomó része angol nyelvű, viszont a magyar címkézésű adathalmazban kiugró számosságú az angol nyelvűként felismert értékelések száma. Ez azzal magyarázható, hogy sok magyar Steam felhasználó tud angolul és valószínűleg jobban preferálta az angol nyelvet az értékelés megírásakor. Ezen problémák kezeléseként a szöveghalmaz előfeldolgozása során mind az angol, mind a magyar nyelvű adathalmazokból csak azon értékeléseket tartottam meg, melyek nyelve megegyezik a felhasználó által megadott nyelvvel.

A korpusz előkészítésének fontos lépése a szövegek hosszának kiértékelése, mivel a BERT tanításakor egy maximális szekvenciahosszt kell beállítani. Ha a megadott érték túl nagy, az a bemenetek felesleges padding-olásához vezethet. A szöveghosszok nagy variabilitása miatt a szövegrekordokat hosszuk alapján vödröztem oly módon, hogy 50 egyenlő szélességű vödröt alakítottam ki. A kapcsolódó hisztogramokat a 10. és a 11. ábrák mutatják be (az y tengely logaritmikus nagyságrendű).



10. ábra Magyar adathalmaz értékeléseinek hosszai 50 egyenlő méretű vödörbe rendezve



11. ábra Angol adathalmaz értékeléseinek hosszai 50 egyenlő méretű vödörbe rendezve

A 10. és 11. ábrákon hasonló eloszlás látható a két adathalmaz tekintetében. Az angol szöveghalmazban található 2 kiugró érték, ezeket megvizsgálva azt tapasztaltam, hogy tényleges tartalmat nem hordoznak, hanem csak sok újsorjelet. Így ezen értékelések felfoghatóak csak zajnak. Ahogy az előkészítések fejezetnél látható lesz, ezeket töröltem. Az értékelési szövegek hosszát tekintve a két nyelvre vonatkoztatott eloszlás hasonló, mivel mindkét szöveghalmaz nagyon sok rövidebb értékelést tartalmaz (pl. „Nice!”, „nagyon jó”), viszont hosszú, és részletes értékelések is 10-es nagyságrendekben szerepelnek bennük.

3.2. Kutatási módszertan

A kutatás során az összegyűjtött adathalmazokon olyan BERT alapú bináris osztályozásokat valósítottam meg, amelyeket különböző előfeldolgozásokkal, különböző modell-összeállításokban hajtottam végre és teszteltem le.

3.2.1. A szöveghalmazok előkészítése

Az osztályozási feladat megvalósításához szükséges volt a tanító (tréning) és a kiértékelő (validation) adathalmazok kialakítása, melyek különféle előkészítő tevékenységek végrehajtását is igényelték.

Az alkalmazott lépések a következők voltak:

1. **Az értékelések szűrése a szövegek hossza szerint:** ahogy a 3.1. alfejezetben is említettem, a leggyűjtött szöveghalmazban változóak a

szövegek hosszai. Rengeteg pár szavas értékelést is tartalmaznak, viszont ezek túl egyszerűek (pl.: „Jó”). Továbbá, a BERT fix hosszú bemenete miatt célszerű a hasonló méretű értékeléseket együttesen feldolgozni, így a túl hosszú értékeléseket is érdemes kiszűrni. Ezek alapján a 25 és 500 karakterhossz közötti értékeléseket találtam legmegfelelőbbnek, s csak ezeket tartottam meg.

2. **Nyelvek szerinti szűrés:** a korábban bemutatott 8. és 9. ábra a tényleges nyelvek eloszlását mutatja be az alapadathalmazokban. Mint korábban is jeleztem, a magyar alapadathalmazból csak a ténylegesen magyar értékeléseket tartottam meg, az angolból pedig csak az angolokat.
3. **Tanítási és kiértékelési halmazra bontás:** az adatok 80%-át tanítási halmazba, a maradék 20%-át pedig kiértékelési halmazba soroltam be. A tréning és validation halmazok kialakítása véletlenszerűen történt, amely nem vette figyelembe a leggyűjtött szövegek sorrendjét.

Ezekkel a lépésekkel elkészült két-két minimális előfeldolgozású tanító és validációs szöveghalmaz, melyeket a következőkben nyelvük alapján **hun** és **eng** előtag jelölésekkel láttam el.

A 6. és 7. ábrán látható volt, hogy az alapadathalmazok kiegyensúlyozatlanok. A kutatás során szeretném megvizsgálni, hogy hogyan befolyásolja az osztályozás pontosságát az adathalmaz kiegyensúlyozása. Emiatt véletlen mintavételezéssel 3 olyan adathalmazt hoztam létre, melyekben kiegyensúlyoztam vagy a tanuló adathalmazt, vagy a kiértékelő adathalmazt, vagy mindkettőt. Az így létrejött 3 adathalmaz elnevezése rendre a következő: **hun_balanced_onlytraining**, **hun_balanced_onlyeval** és **hun_balanced**.

Mint láhattuk, az angol adathalmaz mérete sokkal nagyobb, mint a magyar nyelvű szöveghalmazé. Mivel a neurális hálózat tanulását jelentős mértékben befolyásolhatja a tanításhoz felhasznált adathalmaz mérete, ezért a kutatásban elsősorban olyan összehasonlításokat szeretnék végezni, ahol a modellek tanítása azonos méretű adathalmazokon történik. Ezen cél megvalósításának érdekében az alap angol szöveghalmazt lekicsinyítettem a magyar szöveghalmaz méretére az előzőekben említett kiegyensúlyozási szempontok figyelembevételével. Ezt a

névkonvenció során „x1” utótaggal jelöléssel láttam el. Az így létrejött adathalmazok a következők: **eng_x1_balanced_onlytraining**, **eng_x1_balanced_onlyeval** és **eng_x1_balanced**.

Másrészt, szintén az angol adathalmaz mérete miatt, lehetőségem nyílt megvizsgálni, hogy javul-e az osztályozási modell eredménye amennyiben növelem a tanulóhalmaz méretét. Emiatt 8 további adathalmazt hoztam létre, melyekben lecsökkentettem az angol szöveghalmaz méretét a magyar méretével azonosra, annak 1 századára, 5 századára, tizedére, felére, kétszeresére, ötszörösére, és tízszeresére. Ezen szöveghalmazok neve rendre **eng_x1**, **eng_x0.01**, **eng_x0.05**, **eng_x0.1**, **eng_x0.5**, **eng_x2**, **eng_x5**, **eng_x10**. Továbbá, mivel az egyszerűsénél kisebb adathalmazok a magyar adathalmazon is előállíthatóak, ezért ezeket a magyar esetén is létrehoztam, ezek a következők: **hun_x0.01**, **hun_x0.05**, **hun_x0.1**, **hun_x0.5**.

A kutatáshoz előkészített szöveghalmazok fő paramétereit az 1. táblázat foglalja össze.

1. Táblázat A kutatáshoz használt adathalmazok jellemzői

<i>Szöveghalmaz nyelve</i>	<i>Értékelések száma</i>	<i>Pozitív („Ajánlott”) értékelések száma</i>	<i>Negatív („Nem ajánlott”) értékelések száma</i>
hun	85 473	73 327	12 146
hun_balanced	24 291	12 145	12 146
hun_balanced_onlyeval	24 340	19 163	5177
hun_balanced_onlytraining	24 294	13 907	10 387
eng_x1_balanced	31 389	15 694	15 695
eng_x1_balanced_onlyeval	31 670	23 760	7910
eng_x1_balanced_onlytraining	32 472	18 271	14 201
eng_x0.01	854	717	137
eng_x0.05	4273	3494	779
eng_x0.1	8547	6941	1606
eng_x0.5	42 736	34 700	8036
eng_x1	85 473	69 203	16 270

<i>eng_x2</i>	170 946	138 898	32 048
<i>eng_x5</i>	427 365	347 010	80 355
<i>eng_x10</i>	854 730	694 559	160 171
<i>hun_x0.01</i>	854	736	118
<i>hun_x0.05</i>	4273	3694	579
<i>hun_x0.1</i>	8547	7347	1200
<i>hun_x0.5</i>	42 736	36 659	6 077

A táblázat által bemutatott adathalmazokat különböző BERT modellek tanításához és kiértékeléséhez használtam fel, melyeket a következőkben részletezek.

3.2.2. Alkalmazott modellek

Mivel angol és magyar szöveghalmaz áll rendelkezésemre a kiértékeléshez, ezért az angol adathalmazon előtanított BERT-Base modellt és a magyaron előtanított huBERT-et alkalmaztam a finomításokhoz. Ahogy 2.3.4 fejezetben korábban említettem, ezeknek 2 verziója is van, a cased és az uncased (magyarnál uncased helyett lowercased) modellek. Kutatásomban mindegyiket megvizsgáltam, így összesen 4 modellt értékeltem ki, melyek a következők: cased angol BERT-Base, uncased angol BERT-Base, cased huBERT és lowercased huBERT.

Mivel az 1. táblázatban látható adathalmazok nyers szövegformátumúak, ezért szükséges lépés ezen szöveghalmazok tokenizálása, mely a BERT modell finomítása előtt történik meg. Ez modelfüggő, más lépések szükségesek a cased, uncased és lowercased modellekhez. A tokenizálást az eredeti BERT implementáció alapján végeztem el, és a következő lépésekből állt:

1. Szöveg normalizálása: ebben a lépésben a szövegek előkészítése modellenként eltérő. Cased modellek esetén ez a lépés kimaradt, viszont uncased modellek esetén, a szöveg kisbetűssé alakítása szükséges volt, a diakritikus jeleket pedig le kellett cserélni. A lowercased modell esetén pedig csak kisbetűssé alakítás történt ebben a lépésben. Emiatt használja ezt a verziót huBERT, hiszen a magyar nyelvben szemantikailag fontosak a diakritikus jelek.

2. Központoszási jelek leválasztása: ezen lépésben a központoszási jelek köré szóköz került, így saját tokenként szerepeltek.
3. WordPiece tokenizálás: a tokenek itt véglegesültek, whitespace karakterek jelentették a tokenek közti határokat. Az így kinyert tokeneken WordPiece tokenizálást [24] hajtottam végre, mely karakter alapon alszavakat talált meg a korpuszban. Így könnyebb lehet például ragozott szavak felismerése. Az alkotott alszavak speciális karakterrel kezdődtek, így lehetett visszaalakítani a felbontott szavakat, mint például: készít = kész + ##ít.

3.2.3. A modellek paraméterhangolása

A kutatás folyamán a megfelelő osztályozási eredmények elérésének érdekében kiemelten fontos a neurális hálózat hiperparamétereinek megfelelő beállítása. Emiatt a kutatás első lépéseként hiperparaméter-hangolást hajtottam végre. A BERT hiperparaméterei közül Jacob Devlin-ék is kiemelik a legfontosabbakat [8], ezek a batch méret, a tanulási ráta (learning rate) és az epoch-ok száma, s így én is ezen paraméterek hangolását végzem el. (A tanulási ráta az Adam algoritmus paramétere, ez szolgál a BERT optimalizálójaként weight decay-el kiegészítve.) Ezekhez a szerzők javasolt értékeket is megadtak, amik jól működhetnek különböző feladatokon. Ezek a következők:

- Batch méret: 16, 32
- Tanulási ráta: 5e-5, 3e-5, 2e-5
- Epochok száma: 2, 3, 4

Kutatásaim során a tanulási ráta értékkészletét kiegészíttem az 1e-5 és 6e-5 értékekkel. A tanulási ráta és a batch méret kombinációit grid search algoritmussal értékeltem ki. Emellett epoch-onként kiértékelést végeztem, így azt is megvizsgáltam, hogy melyik epoch-ban volt optimális a veszteségfüggvény. Minden futtatást 4 epoch-ig végeztem, az ajánlás szerint. A kiértékeléshez alkalmazott veszteségfüggvényt a 3-as képlet mutatja be a következő fejezetben.

A paraméterhangolás mellett, a cased és uncased/lowercased modellek közötti választási döntést is ebben a kutatási fázisban hoztam meg úgy, hogy mindkét

változatra elvégeztem a hiperparaméter-hangolást, majd a jobb veszteségfüggvény értékű modellt választottam. Ahogy már említettem, a nyelvi sajátosságokat figyelembe véve ebben a kutatási fázisban 4 modellt alkalmaztam, a cased és uncased angol BERT-Base-t, és a cased, lowercased huBERT-et. A tokenizálást minden esetben a modellnek megfelelő módon végeztem el, azaz cased, uncased, lowercased modellekhez rendre cased, uncased és lowercased tokenizálást hajtottam végre.

3.2.4. *Tesztscenáriók*

A hiperparaméter-hangolást követően a következő forgatókönyvcsoportok szerinti tesztek hajtottam végre (az előző fázisban meghatározott hiperparaméter-beállítások mellett):

- **Szenárió I: Modellek bilingualitása:** a *hun* és *eng_x1* adathalmazokon magyar és angol nyelvre betanított BERT modellekkel végeztem osztályozást (huBERT és angol BERT-Base). Ezzel lehetőségem volt két azonos méretű, de eltérő nyelvű szövegeket tartalmazó adathalmazon megvizsgálni, hogy az egyes modellek milyen teljesítményt nyújtanak a saját és egy idegen nyelvű szöveghalmaz bináris osztályozása során.
- **Szenárió II: Kiegyensúlyozás hatása:** A második csoportba tartozó tesztesetek célja a kiegyensúlyozott és nem kiegyensúlyozott adathalmazok közti különbségek kiértékelése volt. Ehhez az összes *hun_* és az egyszeres méretű *eng_x1_* kezdetű szöveghalmazokat használtam fel. A modellek pedig a magyar adathalmazok esetében a huBERT, az angolok adathalmazok esetében pedig az angol BERT-Base voltak.
- **Szenárió III: A szöveghalmaz méretének hatása az osztályozás teljesítményére:** az angol BERT-Base modellen az *eng_x0.01*, *eng_x0.05*, *eng_x0.1*, *eng_x0.5*, *eng_x1*, *eng_x2*, *eng_x5* és az *eng_x10* adathalmazokat, a magyar huBERT modellen pedig a *hun_x0.01*, *hun_x0.05*, *hun_x0.1*, *hun_x0.5* és a *hun* adathalmazokat futtatva azt

vizsgáltam, hogy milyen hatása van az adathalmaz méretének az osztályozás pontosságára.

- **Szenárió IV: Cased, Uncased és Lowercased esetek vizsgálata:** a hiperparaméter-hangolás során elvégzett futásoknál csak a veszteségfüggvényt használtam fel és nem részleteztem a különbségeket, emiatt ebben a szenárióban más metrikákkal és szempontokkal is kiértékelem a 4 modellt, mindet a saját nyelvű adathalmazán.

3.2.5. Kiértékelés módszertana

A kísérletek kiértékeléséhez több kvantitatív mértéket is használtam. A tanításhoz is használt veszteségfüggvényt, az F1-mértéket, pontosságot kiegyensúlyozott pontosságot, és az ezek komponenseinek tekinthető precizitás, szenzitivitás és specificitás mértékeket.

A veszteségfüggvény (*loss*) kiszámítási módját a következő képlet mutatja be:

$$\text{loss}(\mathbf{X}) = A \left(-\sum (\mathbf{l} * \text{LogSoftmax}(\mathbf{X})) \right) \quad (3)$$

ahol $\mathbf{X}$ a BERT utolsó rétegének kimenete, A egy mátrix elemeinek számtani közepét kiszámító függvény, $\mathbf{l}$ az osztályozás címkéinek one-hot-enkódolt vektora, a *LogSoftmax* függvényt pedig a következőképpen definiáljuk:

$$\text{LogSoftmax}(x_i) = \log \left(\frac{\exp(x_i)}{\sum_j \exp(x_j)} \right) \quad (4)$$

ahol x_i a bemeneti vektor i . eleme, így elemenként alkalmazva a *LogSoftmax* függvényt egy új vektort kapunk kimenetnek.

Mivel a BERT kimenete 0 és 1 közötti valószínűségi érték, ezért a *loss* függvény kimenetének tartománya $[0, \infty)$, ahol 0 teljes egyezést jelent az elvárt kimenet és a kapott kimenet között.

Az osztályozás szempontjából a kimenetek a következő csoportokba sorolhatóak: valós pozitív (TP), valós negatív (TN), álpozitív (FP) és álnegatív (FN) esetek. A valós csoportokban a BERT kimenete megegyezik az elvárt kimenettel,

viszont az ál- csoportokban nem. Ezen csoportoknak a számossága alapján megállapítható, hogy mennyire volt jó az osztályozás.

A kiértékelés során a precizitás (más néven pozitív prediktív érték), a szenzitivitás (más néven valódi pozitív ráta), az F1-mérték, és a pontosság minőségi mutatókat határoztam meg. Az osztályozás precizitása a következőképpen határozható meg:

$$precizitás = \frac{TP}{TP+FP} \quad (5)$$

Az osztályozás szenzitivitásának meghatározása:

$$szenzitivitás = \frac{TP}{TP+FN} \quad (6)$$

Az F1-mérték a precizitás és a szenzitivitás harmonikus közepeként adódik:

$$F_1 = 2 \cdot \frac{precizitás \cdot szenzitivitás}{precizitás+szenzitivitás} \quad (7)$$

Az F1-mérték 0 és 1 közötti eredményt ad, ahol 1 jelenti a teljes egyezést, így a magasabb F1 értékkel rendelkező futások elérése a cél. A precizitás és a szenzitivitás értékei szintén a $[0, 1]$ tartományból vesznek fel értékeket, s esetükben is az 1 érték jelenti a legjobb értéket.

Továbbá, az osztályozás pontosságát is meghatároztam a következő módon:

$$pontosság = \frac{TP+TN}{TP+TN+FP+FN} \quad (8)$$

Vagyis a pontosság a helyesen azonosított esetek, és az összes eset számának az aránya. Ez viszont, jellegéből adódóan félrevezető metrika lehet kiegyensúlyozatlan adathalmazok esetén.

A kiegyensúlyozatlan adathalmazok értékeléséhez sok egyéb metrika is használható. Jelen kutatásban a kiegyensúlyozott pontosságot alkalmaztam, mely a szenzitivitás és a specificitás (más néven valódi negatív ráta) számtani közepe. A szenzitivitás kiszámolását már bemutattam a 6. képletben, a specificitás képlete pedig a következő:

$$specificitás = \frac{TN}{TN+FP} \quad (9)$$

Így, az előbbi két mértékből kiszámítható a kiegyensúlyozott pontosság:

$$\text{kiegyensúlyozott pontosság} = \frac{\text{specificitás} + \text{szenzitivitás}}{2} \quad (10)$$

3.2.6. A kutatás technikai háttere

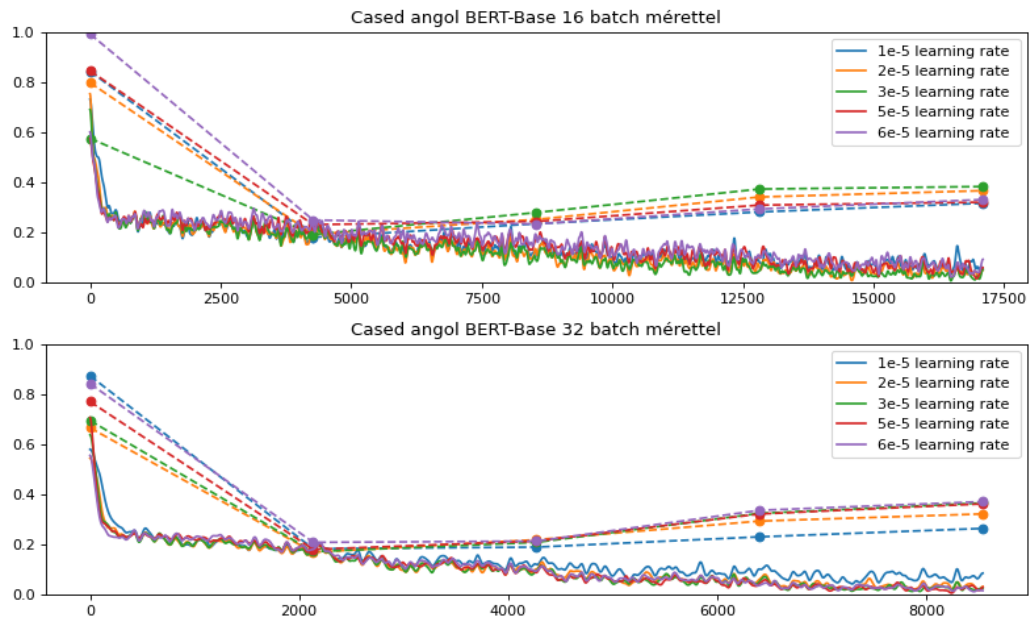
A modellek futtatásához és kiértékeléséhez a BERT alkotói által készített nyílt forráskódú implementációt használtam [25], kisebb átalakításokkal. Az eredeti implementáció TPU-ra (Tensor Processing Unit) lett fejlesztve, mely a Google által specifikusan neurálishálózat-tanításra fejlesztett hardvertípus. Ennek előnye, hogy nagy minibatch méreteknél sokkal gyorsabban végzi a futtatást. A kísérletekhez egy Nvidia Titan V GPU-t, 8 magos CPU-t, és 32GB RAM-ot tartalmazó hardvert használtam. Így, mivel a munkához egy GPU állt rendelkezésre, ezért szükség volt az eredeti BERT kód átírására GPU kompatibilissé. Ezen technológiai részletek a dolgozat jellege miatt jelen munkában nem kerülnek részletezésre.

4. Eredmények

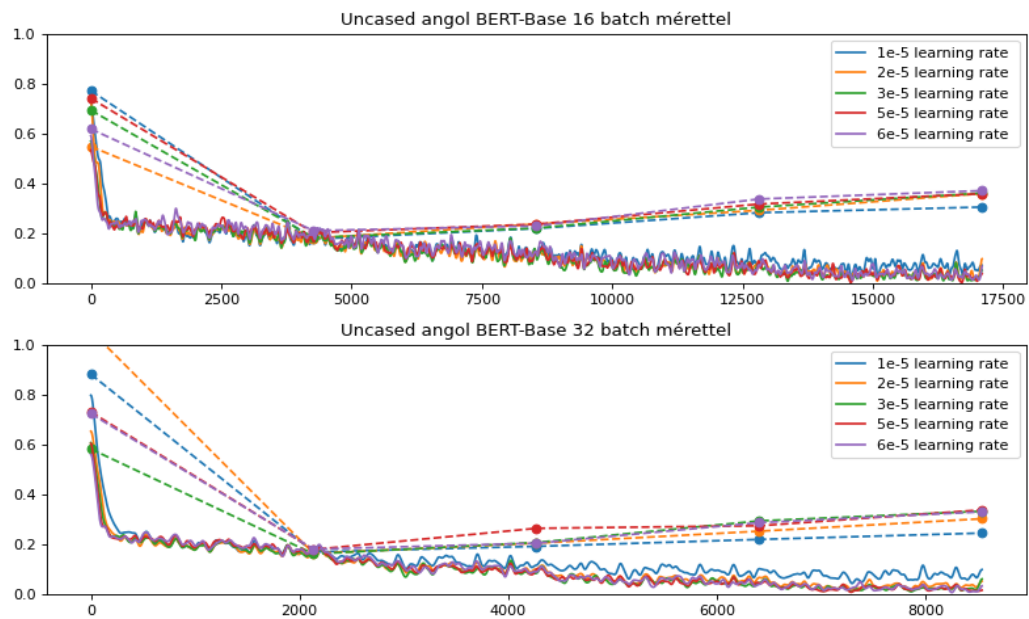
4.1. A BERT modellek hiperparaméter-hangolása és eredményei

A hiperparaméter-hangolás célja a megfelelő tanulási ráta, batch méret és epoch szám meghatározása. A hiperparaméter-hangolást az angol BERT-Base modell esetében a cased és uncased, a huBERT modell esetében pedig cased és lowercased változatok esetén is elvégeztem azon célból, hogy megtaláljam a legideálisabb modell és hiperparaméter-beállítás kombinációkat a további kísérletek elvégzéséhez.

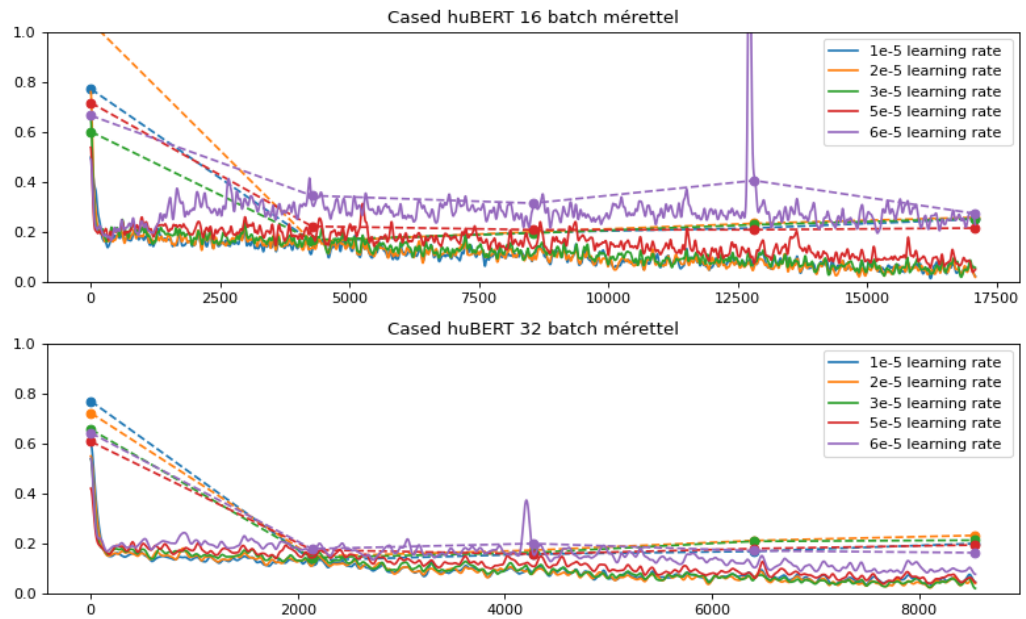
A kis méretű minibatch tanítás miatt, a tanítási fázisban az egyes lépésekhez tartozó loss értékek (lásd 3. egyenlet) nagyban eltérhetnek, nem garantált a loss érték folyamatos csökkenése. Így, hogy jobban látható legyen a tanítási hiba (*training loss*), egy 1D Gauss szűrővel (20-as szigma értékkel) simítottam azt. A modellek hibáját (*evaluation loss*) az egyes epoch-ok végén a kiértékelő szöveghalmaz felhasználásával határoztam meg. Az eredményeket a 12, 13, 14 és 15-ös ábrák mutatják be. Az ábrákon a Gauss szűrővel simított training loss értékek folyamatos vonallal, az evaluation loss értékek pedig szaggatott vonallal vannak ábrázolva. Bár az evaluation loss értékek csak az epochok végén lettek meghatározva, értéküket a trendek szemléltetése és a jobb átláthatóság végett összekötöttem (szaggatott vonallal). A 40 futás eredményét nyelvük szerint, majd azon belül case-ing (uncased, cased, lowercased) és batch méret szerint különítettem el és ábrázoltam.



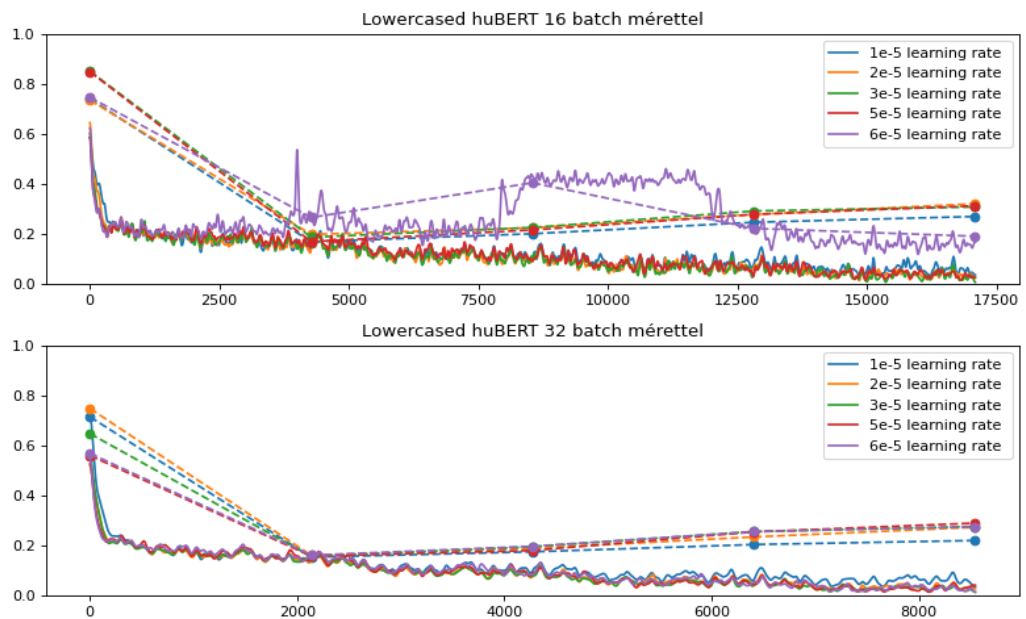
12. ábra Hiperparaméter-hangolás lépésenkénti training loss, és epochonkénti evaluation loss értékei, külön az egyes futásokra cased angol BERT-Base esetén.



13. ábra Hiperparaméter-hangolás lépésenkénti training loss, és epochonkénti evaluation loss értékei, külön az egyes futásokra uncased angol BERT-Base esetén.



14. ábra Hiperparaméter-hangolás lépésenkénti training loss, és epochonkénti evaluation loss értékei, külön az egyes futásokra cased huBERT esetén.

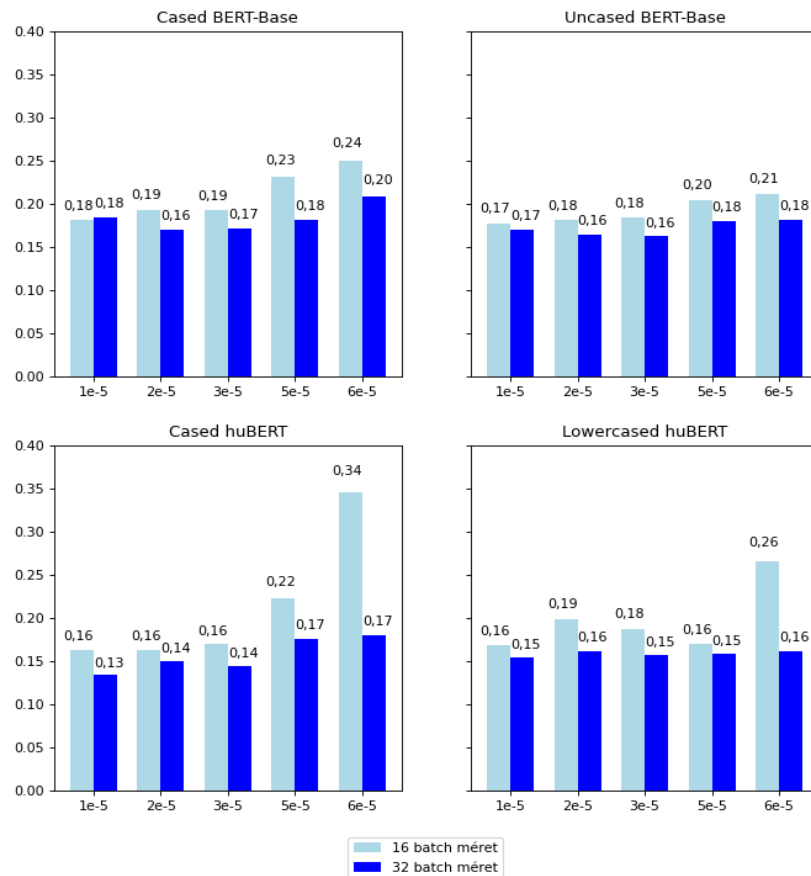


15. ábra Hiperparaméter-hangolás lépésenkénti training loss, és epochonkénti evaluation loss értékei, külön az egyes futásokra lowercased huBERT esetén.

Az eredményeket elemezve láthatjuk, hogy szinte mindegyik tanításra jellemző, hogy a túltanulás már a 2. epoch-nál fellép, hiszen az 1. epoch-ban a legjobb a futások evaluation loss értéke, utána pedig ez az érték növekszik. Ez nagyrészt a

transfer learning-nek köszönhető, hiszen a finomhangolás során általában kis epoch számmal, gyorsan megtanulható egy-egy feladat. Az, hogy ez kisebb az ajánlott epoch számoknál, azt jelezheti, hogy a használt adathalmazok bináris osztályozási feladata nem jelent komoly kihívást a modellnek.

A további általános tulajdonságjegyek (hasonlóságok/különbözőségek) észrevételét nehezíti, hogy nagyon hasonló eredményeket értek el az eltérő tanulási rátával tanított modellek. Mivel látható, hogy szinte mindegyik esetben az 1. epoch-nál a legalacsonyabb az evaluation loss értéke, ezért elegendő csak az itteni értékeket megvizsgálni. Ez látható a 16. diagrammon, az evaluation loss pontos értékeit pedig az 1.sz Melléklet tartalmazza.



16. ábra Hiperparaméter-hangolás futások 1. epoch-jának evaluation loss értékei a tanulási ráta hiperparaméterének függvényében.

A 16. ábra x tengelyén a tanulási ráta, az y tengelyen pedig az adott tesztet 1. epoch-jának evaluation loss értékei szerepelnek, minden esetben a 0 és 0,4 közötti

tartományban ábrázolva. Csak a 16 batch méretű tesztek tekintve szinte minden esetben az látható, hogy a tanulási ráta növelésével a veszteségfüggvény értéke is növekszik. Ennek oka az lehet, hogy nagyobb tanulási ráta értékeknél nehezebben konvergál a modell az optimumra. Láthatjuk továbbá, hogy egy esetet kivéve, a 32 batch méretű tesztek mindenhol jobb eredményt értek el, mint a 16 batch méretűek (a különbség azonban itt minimális). Ez általánosságban is jellemző, hogy a batch méret növelésével az eredmény is javítható, az egyetlen, ami gátolja a batch méret növelését, az a hardver (a mi esetünkben a 64-es batch méret már nem fért memóriába). A cased és uncased/lowercased modell közötti különbségek ezen az ábrán nem feltűnőek, ezért ezeket a 4-es scenárióban részletezem.

A hiperparaméter-hangolás eredményeképpen a magyar és angol BERT modellekhez külön-külön választottam ki a legjobb hiperparaméter értékeket, melyek a következők lettek:

- huBERT:
 - tanulási ráta= $1e-5$,
 - batch méret=32
 - és a cased modell alkalmazása.
 - Ezen hiperparaméter-beállítások és modell alkalmazása mellett a modell evaluation loss értéke: 0,1338.
- angol BERT-Base:
 - tanulási ráta= $3e-5$,
 - batch méret=32,
 - és az uncased modell alkalmazása.
 - Ezen hiperparaméter-beállítások és modell alkalmazása mellett a modell evaluation loss értéke: 0,1623.

4.2. Tesztelési scenáriók eredményei

A következő alfejezetekben bemutatom a tesztszenáriók eredményeit, melyek mind a hiperparaméter-hangolás során elért, az előbb említett legjobb

hiperparamétereket alkalmazzák. Ahol nincs említve, ott a huBERT-nél a cased modell, és az angol BERT-Base-nél pedig az uncased modell az értendő.

4.2.1. Szcenárió I: Modellek bilingualitása

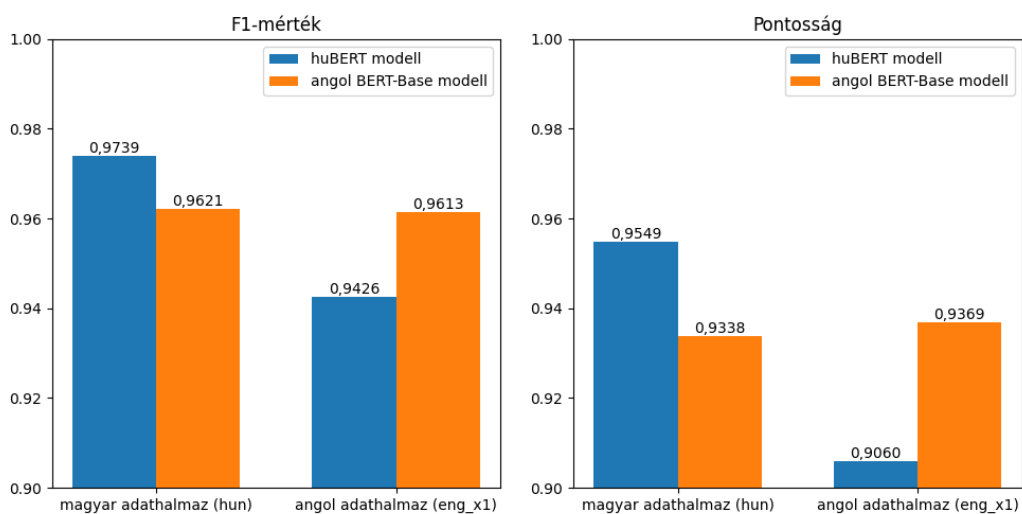
A huBERT és az angol BERT-Base mind a saját nyelveiken lettek előtanítva, ezért az elvárható eredmény, hogy mindegyik modell a saját nyelvű adathalmazán teljesít jobban. Viszont kérdéses, hogy mekkora mértékű az eltérés egy idegen nyelvű szöveghalmaz felhasználása során, tehát megtudtak-e tanulni egy új nyelvet a modellek. Ez függhet attól, hogy az előtanításhoz és a finomhangoláshoz használt szöveghalmazok nyelve mennyire hasonlít, illetve, hogy a két modell finomhangolásához használt szöveghalmazok mennyisége és jellege mennyiben különbözik. Kérdés továbbá, hogy melyik teljesít jobban a saját nyelvi környezetében. A kiértékeléshez tartozó keveredési mátrix a 2. táblázatban, a minőségi mutatószámok értékei a 3. táblázatban, az F1-mérték és a pontosság grafikus megjelenítése pedig a 17. ábrán látható. Az egyértelműség kedvéért fontos megjegyezni, hogy a kiértékelések természetesen a szöveghalmazok validation részhalmazán lettek végrehajtva, azért ilyen alacsonyok a gyakorisági értékek a keveredési mátrixban. Az eng_x1 és hun szöveghalmazok validációs részhalmazai egyaránt 17 094 értékelést tartalmazott.

2. táblázat 1. szcenárió BERT modelleinek keveredési mátrixa

	huBERT			angol BERT-Base	
	Jósolt \ Elvárt érték	Ajánlott	Nem ajánlott	Ajánlott	Nem ajánlott
	Ajánlott	13 189	951	13 407	641
Angol szöveghalmaz (eng_x1)	Nem ajánlott	655	2 299	437	2 609
	Ajánlott	14 371	437	14 369	796
Magyar szöveghalmaz (hun)	Nem ajánlott	333	1 953	335	1594

3. táblázat Bilingvális szcenárió modell- és adatpárjainak metrikái

Adat + modell név	evaluation loss	precizitás	szenzitivitás	F1- mérték	pontosság	kiegyensúlyozott pontosság
eng_x1 + huBERT	0,2240	0,9327	0,9526	0,9426	0,9060	0,8300
hun + huBERT	0,1217	0,9704	0,9773	0,9739	0,9549	0,8972
eng_x1 + angol BERT-Base	0,1607	0,9543	0,9684	0,9613	0,9369	0,8856
hun + angol BERT-Base	0,1687	0,9475	0,9772	0,9621	0,9338	0,8220



17. ábra Az angol BERT-Base modell és a huBERT teljesítménye a saját nyelvű és idegen nyelvű adathalmazon az F1-mérték és a pontosság szempontjából értékelve.

A 17. ábrán látható, hogy mindkét modell mindkét nyelv tekintetében jó eredményeket ért az osztályozási feladat megvalósítása során. Az eredmények minden esetben meghaladták a 0,9-es értéket (mind a pontosság, mind az F1 mérték esetében), így, hogy jobban láthatóak legyenek a különbségek, az ábrán csak a 0,9 és 1 közötti tartomány látható. Láthatjuk, hogy a feltetelezés – mely szerint mindegyik szövegen a saját nyelvű modell teljesít jobban – teljesült.

Érdekes megfigyelni, hogy az angol BERT-Base szinte azonos F1 és pontosság értékeket ért el mindkét adathalmazon. Vizsgáljuk meg például az F1-mértéket. Korábban említettem, hogy az F1-mérték egy összegző mérték, a precizitás és a szenzitivitás mértékéből áll össze. Ezeket tekintve a 2. táblázatban, azt láthatjuk, hogy az angol BERT precizitása jobb a saját nyelvű adathalmazán,

viszont szenzitivitása rosszabb. Mindkét esetben hasonló mértékű az eltérés, emiatt kapott ez a modell hasonló F1 értékeket az angol és magyar szöveghalmazon. Az ellentétes irányú eltérés oka, hogy kis mértékben eltérő a pozitív/negatív értékelések aránya a hun és eng_x1 szöveghalmazokban (1. táblázat), mivel véletlenszerűen lettek az értékelések kiválasztva. Pontosabban, az eng_x1 halmazban több negatív értékelés van, mint a hun halmazban, így a modellek is nagyobb valószínűséggel jósolnak negatív értékelést az angol halmaz esetében (ez látható a 3. táblázatban is). A szenzitivitás pedig akkor lesz magas érték, ha a hamis negatív értékelésekből minél kevesebbet eredményez egy modell. Így mivel a magyar adathalmazban kevesebbszer jósolt negatív értékelést az angol BERT, hamis negatív értékelésből is kevesebbet szerzett, így kaphatott jobb szenzitivitást ezen a halmazon. Szintén ez - a magyar és angol adathalmaz közötti kiegyensúlyozásbeli különbség - az oka a pontosság mértékében tapasztalható hasonlóságnak is, hiszen a kiegyensúlyozott pontosságban már jobb értéket kapott az angol BERT angol adathalmazon, mint magyaron.

A modellek bilingvualitásának tekintetében elmondható, hogy mindkét modell kiemelkedően teljesített, hiszen mind a 4 esetben 0,9-nél jobb pontosságot és F1 értéket ért el. Ebből feltételezhetjük, hogy a BERT az előtanítás során egy általánosabb tudást szerez meg, így az előtanítás során nem feltétlenül szükséges alkalmazkodni a „downstream” feladatok nyelvéhez vagy témájához.

Ennek ellenére, a huBERT-ről elmondható, hogy kevésbé tudta megtanulni az angol nyelvet, mint az angol BERT a magyar nyelvet (habár ahogy előbb említettem így is jó eredményt ért el). A legtöbb mértéket tekintve, a huBERT a saját nyelvén jobban teljesített, mint az angol a sajátján, viszont idegen nyelven már az angol BERT ért el jobb eredményeket. Tehát elmondható, hogy a huBERT előtanítása sokkal specifikusabb volt a Steam-es szöveghalmazunk szempontjából, ezért kevésbé bilingvális mint az angol BERT. Viszont ennek előnye, hogy az általa ismert nyelven jobb eredményt tud így elérni egy „downstream” feladatnál.

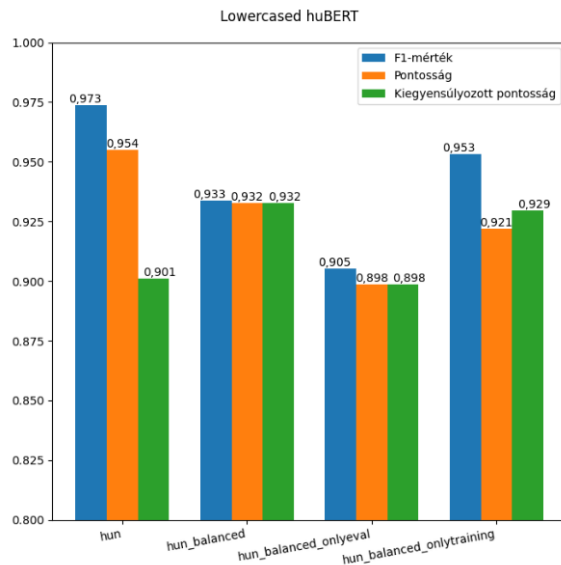
A következő szcenáriókban már nem vizsgálom a modellek idegen nyelvi teljesítményét, minden esetben csak a saját nyelvén fut a modell.

4.2.2. Szenárió II: Kiegyensúlyozás hatása

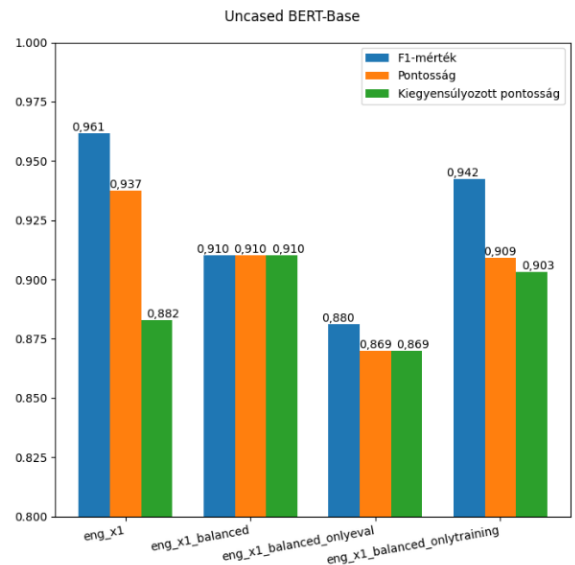
Ahogy már az adathalmazok bemutatásánál említettem, a gyűjtött szöveghalmazok kiegyensúlyozatlanok, mivel bennük sokkal több „Ajánlott” értékelés van, mint „Nem ajánlott”. Emiatt, mind a tanító halmazt, mind a kiértékelőt és egyszerre mindkettőt is kiegyensúlyozva megvizsgáltam, hogy milyen javulás érhető el ilyen jellegű adatelőkészítéssel, illetve milyen hatása van az osztályozásra a kiegyensúlyozatlanságnak. A tanító és kiértékelő halmaz kiegyensúlyozását azért érdemes külön-külön is vizsgálni, mivel a kiértékelés ez utóbbin történik, így az eredményt eltorzíthatja a halmaz esetleges kiegyensúlyozatlansága. Mivel a kiegyensúlyozatlanság ténye a minőségi mutatószámok értékére is jelentős befolyással bír, így a már bemutatott kiegyensúlyozott pontosságot (10. egyenlet) is kiszámítottam az osztályozásokra vonatkozóan. Az eredmények – külön a magyar és angol kiértékelésekre – 4. táblázatban és a 18. és 19. ábrákon tekinthetők meg.

4. táblázat Kiegyensúlyozás Szenárió modell- és adatpárjainak metrikái

<i>Adat+ modell név</i>	<i>evaluation loss</i>	<i>precizitás</i>	<i>szenzitivitás</i>	<i>F1- mérték</i>	<i>pontosság</i>	<i>kiegyensúlyozott pontosság</i>
hun + huBERT	0,1229	0,9718	0,9758	0,9738	0,9549	0,9011
hun_balanced + huBERT	0,1830	0,9378	0,9298	0,9338	0,9326	0,9327
hun_balanced_ onlyeval + huBERT	0,3101	0,8496	0,9683	0,9051	0,8985	0,8985
hun_balanced_ onlytraining + huBERT	0,2007	0,9897	0,9190	0,9530	0,9219	0,9296
eng_x1+ angol BERT-Base	0,1628	0,9527	0,9708	0,9616	0,9373	0,8827
eng_x1_balanced+ angol BERT-Base	0,2260	0,9114	0,9091	0,9103	0,9101	0,9101
eng_x1_balanced_ onlyeval+ angol BERT-Base	0,3058	0,8119	0,9627	0,8809	0,8699	0,8699
eng_x1_balanced_ onlytraining+ angol BERT-Base	0,2128	0,9739	0,9127	0,9423	0,9091	0,9031



18. ábra A huBERT eredményei a magyar kiegyensúlyozott és nem kiegyensúlyozott adathalmazokon.



19. ábra Az angol BERT-Base eredményei az angol kiegyensúlyozott és nem kiegyensúlyozott adathalmazokon.

Az ábrákon láthatjuk, hogy a legjobb pontossága és legjobb F1 értéke a magyar és angol nyelvű osztályozási feladatok esetében is az alap kiegyensúlyozatlan adathalmaz esetében adódik. Ez abból következik, hogy nem csak a pontosság, hanem az F1-mérték is érzékeny a kiegyensúlyozatlanságra, mivel a pozitív címkéjű értékeléseket nagyobb mértékben veszi figyelembe. Éppen ezért fontos tisztában lennünk azzal a ténnyel, hogy ezen mértékek torzítanak nem kiegyensúlyozott adathalmazok esetében. Ebből adódóan a kiegyensúlyozott pontosság mértékének kell fokozott figyelmet szentelni, mivel ezen mérték adja a valószínűbb eredményt. Például a magyar és angol alap kiegyensúlyozatlan adathalmazok esetében is ennek az értéke rosszabb, mint a pontosságé. Az eltérés mértéke 0,0538 a magyar, és 0,0546 az angol esetben (ld. részletesebben a 4. táblázatban). Tehát hiába kaptak ezek a mértékek az alap kiegyensúlyozatlan adathalmaz esetén jobb F1 értéket és pontosságot, a kiegyensúlyozatlanságuk miatt a negatív címkénél rosszabbul teljesítettek, ami a kiegyensúlyozott pontosság értékén is meglátszik.

Viszont, ha a másik három esetet nézzük, ahol a szöveghalmaz valamely része ki lett egyensúlyozva, ott az osztályozás pontossága és kiegyensúlyozott pontossága nagyon közeli értéket kapott. Azokban az esetekben, ahol a kiértékelő halmaz is ki

lett egyenlítve (tehát `balanced` és `balanced_only`val), ott amiatt kapott a két mérték nagyon hasonló értékeket, mivel ezek a mértékek a kiértékelő halmazon elért eredményen számolódnak ki. Az utolsó esetben pedig, ahol csak a tanító halmaz lett kiegyenlítve (`balanced_onlytraining`), a kiegyenlítés hatására kevésbé lett torzított (`biased`) a modell a pozitív értékelések felé, így a pontosság is valósabb értéket mutatott. Ezen megfigyelések alapján levonhatjuk azt a következtetést, hogy abban az esetben, ha mindkét osztály irányába elfogulatlan osztályozót szeretnénk építeni, akkor a tanító halmazt kell kiegyensúlyoznunk, hiszen ezekben az esetekben (tehát `balanced` és `balanced_onlytraining`) jobb kiegyensúlyozott pontosságot kaptunk. Ez az eredmény azt mutatja, hogy habár a kiegyensúlyozással kevesebb adatot kapunk, mégis a modell torzításának javítása nagyobb jelentőséggel bír a modell teljesítményének szempontjából.

A `balanced_only`val esetek, habár nem javítják a modellek torzítását (és így nem javítják az eredményét sem), mégis rávilágítanak arra, hogy nem feltétlenül szükséges olyan mértékeket használni, melyekre nem hat a kiegyensúlyozatlanság. Ezekben az esetekben a tanítóhalmaz kiegyensúlyozatlan, ezért a modell is torzított lesz, viszont mivel a kiértékelő halmaz kiegyensúlyozott, ezért az azon számolt mértékek mégis valós eredményt fognak mutatni. Például a `balanced_only`val eseteknek az F1 értéke rosszabb lett, mint az eredeti, teljesen kiegyensúlyozatlan esetnek.

Összességében a teljes adathalmaz kiegyensúlyozásával, a `huBERT` esetében 0,0316-tal, az angol `BERT` esetében pedig 0,0274-gyel tudtam javítani a modellek kiegyensúlyozott pontosságát. Tehát esetünkben nem nagy mértékben, viszont mindenképp negatív hatása van a kiegyensúlyozatlanságnak a modellek valós teljesítményére.

4.2.3. Szcenárió III: A szöveghalmaz méretének hatása az osztályozás teljesítményére

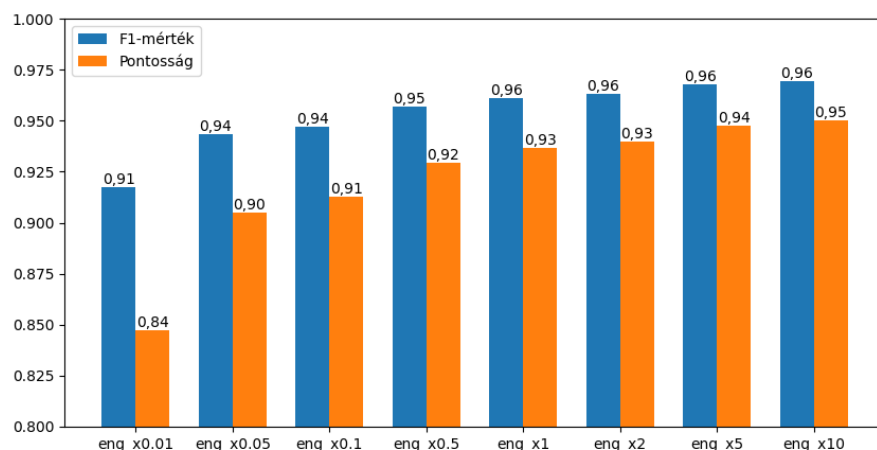
Az angol adathalmaz nagyobb mérete miatt lehetőségem nyílt arra, hogy leteszteljem mekkora és milyen hatása van az adatok mennyiségének az angol BERT-Base-re. Általánosságban elmondható, hogy több adattal jobb eredményeket lehet elérni, így mindenképp javulás várható el az adathalmaz méretének növelésével, viszont kérdéses, hogy ez milyen mértékű. Az eredmények az 5. és 6. táblázatokban és a 20. és 21. ábrákon láthatók.

5. táblázat Méretezési szcenárió metrikái az angol adathalmaz különböző méretein

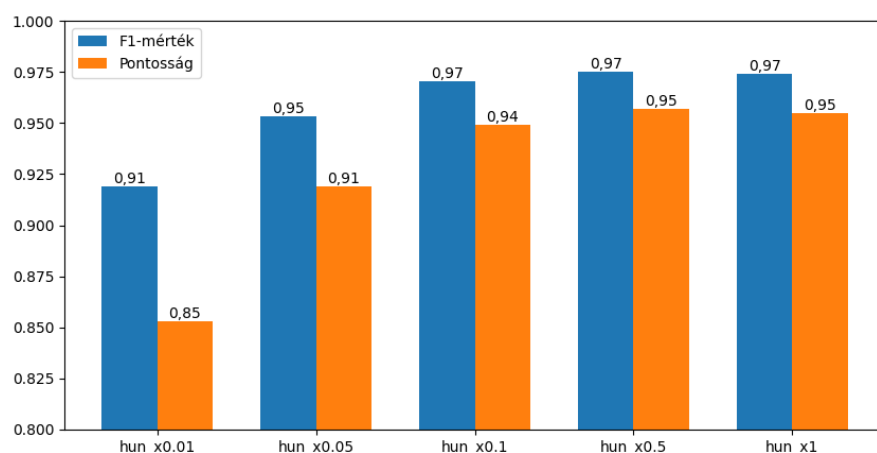
<i>Adat+ modell név</i>	<i>evaluation loss</i>	<i>precizitás</i>	<i>szenzitivitás</i>	<i>F1- mérték</i>	<i>pontosság</i>	<i>kiegyensúlyozott pontosság</i>
eng_x0.01 + uncased angol BERT-Base	0,332	0,847	1,0000	0,9171	0,847	0,5000
eng_x0.05 + uncased angol BERT-Base	0,242	0,9185	0,9699	0,9435	0,9051	0,7926
eng_x0.1+ angol BERT-Base	0,2227	0,9403	0,9532	0,9467	0,9128	0,8449
eng_x0.5+ angol BERT-Base	0,1756	0,9512	0,9622	0,9567	0,9293	0,8747
eng_x1+ angol BERT-Base	0,1607	0,9522	0,9704	0,9612	0,9367	0,8816
eng_tsv_x2+ angol BERT-Base	0,1551	0,9591	0,9672	0,9631	0,9396	0,8923
eng_x5+ angol BERT-Base	0,1347	0,9644	0,9709	0,9676	0,9473	0,9081
eng_x10+ angol BERT-Base	0,1280	0,9645	0,9744	0,9694	0,9502	0,9102

6. táblázat Méretezési szcenárió metrikái, a magyar adathalmaz különböző méretein

<i>Adat+ modell név</i>	<i>evaluation loss</i>	<i>precizitás</i>	<i>szenzitivitás</i>	<i>F1- mérték</i>	<i>pontosság</i>	<i>kiegyensúlyozott pontosság</i>
hun_x0.01 + cased huBERT	0,2988	0,8502	1,0000	0,919	0,8529	0,5535
hun_x0.05 + cased huBERT	0,2	0,9525	0,9538	0,9532	0,9192	0,8273
hun_x0.1 + cased huBERT	0,1547	0,9635	0,9774	0,9704	0,949	0,8794
hun_x0.5 + cased huBERT	0,1142	0,9746	0,9754	0,975	0,9569	0,9077
hun_x1 + cased huBERT	0,1212	0,9705	0,977	0,9737	0,9547	0,8972



20. ábra Az angol uncased BERT-Base modell F1 értéke és pontossága a különböző méretű angol adathalmazokon.



21. ábra A magyar cased huBERT modell F1 értéke és pontossága a különböző méretű magyar adathalmazokon.

Az eredmények azt mutatják, hogy nagyobb adathalmazok esetében valóban jobb lett mind az F1-mérték értéke és a pontossága is az angol BERT-Base-nek. A változás lineáris jellegű, kis mértékű, és nem arányos az adathalmaz méretének változásával. Míg az eng_x10-ben 10-szer annyi értékelés van, mint az eng_x1-ben, addig F1 értéke csak 0,85%-kal, pontossága pedig csak 1,44%-kal javult. Ennek fő oka, hogy a legjobb elérhető F1 értéket – vagyis az 1-et – már a 0,01-szeres adathalmaz (eng_x0.01) is megközelítette. Mindegyik tesztelt esetben 0,9-nél jobb értékeket kaptunk F1-mérték tekintetében és pontosság tekintetében is csak a 0,01-szeres adathalmaz nem érte el a 0,9-es értéket. Így ezeken már nehezebben javít a

modell, hiszen az esetleges hibáit már valószínűleg pár, az átlagostól eltérő értékelés okozhatja. Tehát az adathalmazunk méretének növelésével csak az ilyen speciális esetek előfordulásának a lehetőségét növeljük, az adatok nagy részét viszont már ismeri a modell, és ezeken nem képes újat tanulni. A magyar adathalmazra is ezek a következtetések vonhatóak le, arányokban hasonló változás tapasztalható ebben az esetben is.

A 0,01-szeres modellek kis mérete miatt, az eredményüknek nagy a varianciája, viszont 0,8 alatti F1 értéket és pontosságot így sem kaptam. Habár ez az eredmény kiemelkedőnek mondható, megvizsgálva az 5. és 6. táblázat többi metrikáit, már más kép rajzolódik ki. Az itteni értékek nagy szórásúak, mindkét 0,01-szeres adathalmazon futó modell szenzitivitása 1, viszont kiegyensúlyozott pontossága 0,5 körüli. A szenzitivitás esetében ez akkor történhet meg, ha egy értékelést sem osztályoz hibásan negatívként a modell. Mivel a modellek szinte csak pozitív címkét jósoltak minden értékeléshez, így hibás negatív osztályozásuk sem lehetett, tehát a szenzitivitásuk 1 lett. Így elmondható, hogy a modell 1-es szenzitivitása és ezáltal magas F1 értéke sem mondható jó eredménynek.

Összességében az látható, hogy a modellek már a 0,1-nél nagyobb méretezésű adathalmazokon is jó eredményt értek el, mivel ez a mennyiségű adat is elég reprezentatív volt a teljes halmazra nézve. Az 1. táblázatra visszatekintve láthatjuk, hogy ez a 0,1-szeres adathalmaz 8547 db értékelést tartalmazott, amiből 80%-át, vagyis 6837 esetet használtunk fel tanításra. Az eredményből látható, hogy nem csak az adathalmaz mérete a fontos teljesítmény szempontjából, hanem a változatossága is.

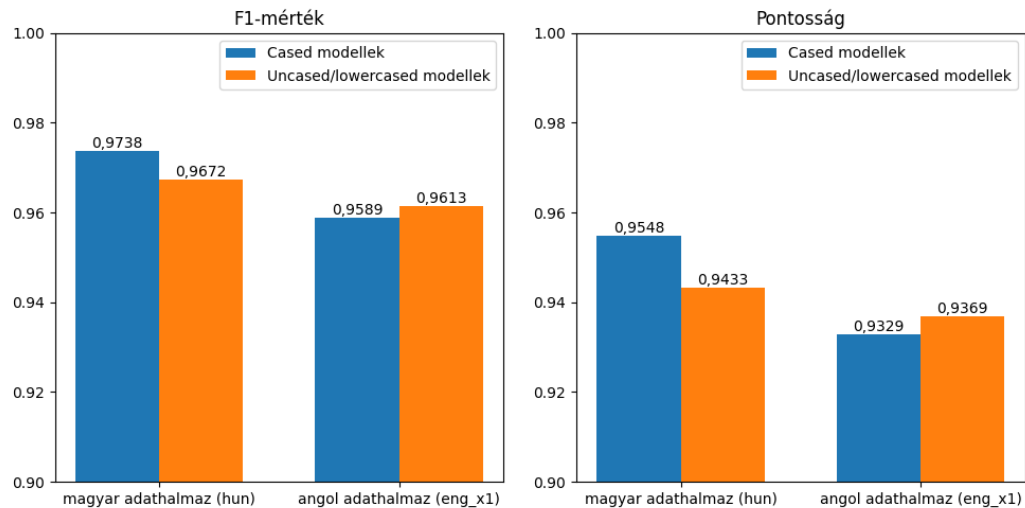
4.2.4. Szcenárió IV: Cased, Uncased és Lowercased esetek vizsgálata

A hiperparaméter-hangolás során már megállapítottam, hogy a modellek 2 változata közül melyik teljesített jobban, viszont nem részleteztem ezt az eredményt. Így ebben a tesztben, csak a hangolás során kiválasztott hiperparaméterekkel teszteltem le, az angol BERT-Base cased és uncased, és a huBERT cased és lowercased verzióját. Az eredmények a 7. táblázatban és a 22. ábrán láthatóak. Mindegyik modell csak a saját adathalmazán futott, így a 22. ábrán

a narancssárga oszlop a magyar esetben lowercased huBERT-et, angol esetben pedig uncased angol BERT-Base modellt jelöl.

7. táblázat Cased, uncased, lowercased szcenárió modell- és adatkéntjainak metrikái

Adat + modell név	evaluation loss	precizitás	szenzitivitás	F1- mérték	pontosság	kiegyensúlyozott pontosság
hun_tsv + cased huBERT	0,1219	0,9723	0,9753	0,9738	0,9548	0,9023
hun_tsv + lowercased huBERT	0,1506	0,9611	0,9734	0,9672	0,9433	0,8657
eng_tsv_x1 + cased angol BERT-Base	0,1685	0,9509	0,9671	0,9589	0,9329	0,8772
eng_tsv_x1 + uncased angol BERT-Base	0,1607	0,9543	0,9684	0,9613	0,9369	0,8856



22. ábra Angol BERT-Base cased/uncased, és huBERT cased/lowercased F1 értéke és pontossága a saját nyelvű adathalmazukon.

Az angol adathalmazon az uncased jobb eredményt ért el, mint a cased modell, mely nem meglepő, hiszen Jacob Devlin és munkatársai is úgy találták [8], hogy csak azokban az esetekben teljesít jobban a cased modell, amikor a kisbetűk/nagybetűk jelentősek a feladat szempontjából (pl.: névelem-felismerés).

Habár a magyar adathalmaz esetében fordított a helyzet, a két eset nem hasonlítható össze, hiszen a huBERT uncased helyett lowercased tokenizálást alkalmaz. Arra, hogy miért teljesít jobban a magyar cased modell, a hipotézisem, hogy a használt magyar adathalmazban fontos szerepe van a nagybetűeknek, mivel egy negatív, agresszívabb hangvételű értékeléshez több nagybetűt használhat a

felhasználó. Összességében nem túl nagyok az eltérések, mely szintén annak köszönhető, hogy nem okozott problémát a BERT-nek a probléma és adathalmaz páros.

Továbbá, megfigyelhető, hogy habár F1 tekintetében közeli értéket kapott a lowercased huBERT és az uncased angol BERT-Base, addig kiegyensúlyozott pontosságban 0,0199-cel jobb eredménye lett az uncased modellnek. Mint említettem, ez a két eset nem feltétlenül összehasonlítható, mégis rávilágítanak arra, hogy egyszerűbb esetekhez akár érdemes lehetne egy uncased huBERT modell teljesítményét megvizsgálni.

Összességében tehát elmondható, hogy Jacob Devlin és munkatársai javaslata az uncased-re valóban igazolódott, viszont a huBERT esetén más szempontokat kellett figyelembe venni, ezért ennek részletesebb megvizsgálása további kutatás témája lehet.

5. Összefoglalás

A transfer learning metodikának hála, az NLP-ben is lehetőség nyílt kis adathalmazokon, kevés erőforrással, jó eredményeket elérni, és ennek része a BERT modell is. Célom annak feltárása volt, hogy milyen kapcsolatban állnak a BERT modellek a bemeneteikkel és hogy milyen mértékben függenek az előtanításukhoz használt adathalmaz nyelvétől, tehát mennyire képesek egy ettől eltérő nyelvet megtanulni és azon feladatot megoldani, illetve milyen paraméterek befolyásolhatják a működésük hatékonyságát. Kutatásom során különböző scenáriókban vizsgáltam a BERT magyar és angol modelljét. Ehhez saját gyűjtésű magyar és angol nyelvű szöveghalmazt használtam, melyeken bináris osztályozási feladatot oldottam meg.

Az 1. scenárióban azt vizsgáltam, hogy az angol nyelven előtanított BERT-Base és a magyar nyelven előtanított huBERT modellek milyen teljesítményt nyújtanak a saját nyelvükön, illetve a számukra idegen nyelven. Az előtanítással megegyező feladatok megoldását tekintve megállapítottam, hogy huBERT speciálisabb, jobban teljesített a saját nyelvén, mint az angol BERT-Base a sajátján, idegen nyelven viszont a huBERT teljesített rosszabbul. Ennek megvizsgálása, hogy mi miatt kevésbé általános a huBERT, további kutatás témája lehet. Emellett bemutattam, hogy a huBERT és az angol BERT-Base modell is képesek voltak a számukra idegen nyelvű adathalmazon is jó eredménnyel elvégezni a bináris osztályozás feladatát (mind az F1-mérték, mind a pontosság mindkét esetben meghaladta a 0,9-es értéket). Így elmondható, hogy mindkét modell valóban képes volt egy új nyelvet megtanulni.

A 2. scenárióban az adathalmazok kiegyensúlyozásának hatásait vizsgáltam. A kísérletek bebizonyították, hogy megéri kiegyensúlyozni legalább a tanító adathalmazt, hiszen így javítható a modell eredménye. Ezentúl az is látható volt, hogy csak a kiegyensúlyozott mértékek tudnak ilyen esetekben valós értéket mutatni, ennek hiányában pedig csak a kiértékelő halmaz kiegyensúlyozásával is kimutathatók az adott modell valós eredményei.

A 3. scenárióban az adathalmaz méretezésének befolyásoló hatását értékeltem ki. Az eredmények arra világítottak rá, hogy a több adaton végzett finomhangolás

csak akkor jár jobb osztályozási eredménnyel, amennyiben azok változatosságot hoznak az eddigi adathalmazba. Ezentúl arra is, hogy kis adatmérték esetén az eredmények nagy varianciája miatt több metrikával is érdemes kiértékelni a modellt, hiszen a modell lehetséges szélsőségei miatt egyes metrikák hamis bizonyosságot adhatnak.

Végül, a 4. scenárióban a cased és uncased/lowercased BERT modellek különbségeit vizsgáltam. Itt látható volt, hogy a BERT alkotóinak ajánlása, miszerint az uncased modell általában jobb eredményeket ad, valóban teljesült az angol nyelvű BERT-Base modell esetében. Viszont a magyar nyelvű huBERT esetén, a cased modell hozta a jobb eredményeket. A lowercased modelljét pedig az angol uncased modell túlteljesítette. Ezen két eredmény okainak vizsgálata további kutatás témája lehet, akár annak megvizsgálásával is, hogy egyszerűbb feladatokhoz jobban tud-e teljesíteni egy uncased huBERT modell.

Munkám folytatásaként olyan szövegfeldolgozásokkal szeretnék foglalkozni, ahol a BERT modellek egy speciális szakterületi feladatot oldanak meg. Példaként említeném az egészségügyet, ahol a szaknyelv sajátossága miatt hasonló probléma állhat elő, mint amit jelen munkámmal modelleztem. Vagyis az előtanított BERT modell egyáltalán, vagy csak kis mértékben ismeri azt a nyelvet, amelyen feladatot kell megoldania. Gondoljunk például egy egészségügyi dokumentumra, amelyet magyar orvosok készítenek magyar nyelven, annak megértése azonban egy egészségügyben nem jártas személy számára nem egyszerű feladat. Mivel a huBERT modellt csupán 2020-ban publikálták, ezért ilyen szakterületspecifikus modell még nem létezik magyar nyelven. A közeljövőben érdemes megvizsgálni, hogy milyen nehézségek árán hozható létre például egy egészségügyi szakterületspecifikus magyar BERT modell, illetve milyen teljesítményt nyújt a huBERT egy ilyen adathalmazon.

Irodalomjegyzék

- [1] Baroni, M., Dinu, G., Kruszewski, G. (2014). Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Volume 1: Long Papers, pp. 238-247.
- [2] Wang, B., Wang, A., Chen, F., Wang, Y., Kuo, C. C. J. (2019). Evaluating word embedding models: methods and experimental results. *APSIPA transactions on signal and information processing*, 8. E19.
- [3] <https://www.tensorflow.org/tutorials/representation/word2vec> (Utoljára meglátogatva: 2021.04.23.)
- [4] Mikolov, T., Chen, K., Corrado, G., Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.
- [5] Pennington, J., Socher, R., Manning, C. D. (2014). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532-1543.
- [6] Bojanowski, P., Grave, E., Joulin, A., Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5, 135-146.
- [7] Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P. (2016). Squad: 100,000+ questions for machine comprehension of text. arXiv preprint arXiv:1606.05250.
- [8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: <https://www.aclweb.org/anthology/N19-1423>

- [9] Bahdanau, D., Cho, K., Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.
- [10] Sutskever, I., Vinyals, O., Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Advances in neural information processing systems, pp. 3104-3112.
- [11] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser L., Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems, pp. 5998-6008.
- [12] Radford, A., Narasimhan, K., Salimans, T., Sutskever, I. (2018). Improving language understanding by generative pre-training.
- [13] Taylor, W. L. (1953). "Cloze procedure": A new tool for measuring readability. *Journalism quarterly*, 30(4), 415-433.
- [14] Nemeskey, Dávid Márk. Natural Language Processing Methods for Language Modeling. PhD Thesis, Eötvös Lóránd University, 2020.
- [15] Xiao, H. (2020). Hungarian layer: A novel interpretable neural layer for paraphrase identification. *Neural Networks*, 131, 172-184.
- [16] Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., & Bowman, S. R. (2018). GLUE: A multi-task benchmark and analysis platform for natural language understanding. arXiv preprint arXiv:1804.07461.
- [17] Jwa, H., Oh, D., Park, K., Kang, J. M., & Lim, H. (2019). exBAKE: automatic fake news detection model based on bidirectional encoder representations from transformers (bert). *Applied Sciences*, 9(19), 4062.
- [18] Li, F., Jin, Y., Liu, W., Rawat, B. P. S., Cai, P., & Yu, H. (2019). Fine-tuning bidirectional encoder representations from transformers (BERT)–based models on large-scale electronic health record notes: an empirical study. *JMIR medical informatics*, 7(3), e14830.

- [19] Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W., & Jiang, P. (2019, November). BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In Proceedings of the 28th ACM international conference on information and knowledge management (pp. 1441-1450).
- [20] <https://store.steampowered.com> (Utoljára meglátogatva: 2021.04.23.)
- [21] <https://partner.steamgames.com/doc/store/getreviews> (Utoljára meglátogatva: 2021.04.23.)
- [22] Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., Mikolov, T. (2016). Fasttext. zip: Compressing text classification models. arXiv preprint arXiv:1612.03651.
- [23] Joulin, A., Grave, E., Bojanowski, P., Mikolov, T. (2016). Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759.
- [24] Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., et al. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv preprint arXiv:1609.08144.
- [25] <https://github.com/google-research/bert> (Utoljára meglátogatva: 2021.04.23.)
- [26] Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., & Kang, J. (2020). BioBERT: a pre-trained biomedical language representation model for biomedical text mining. Bioinformatics, 36(4), 1234-1240.

Mellékletek

1. sz melléklet

Hiperparaméter-hangolás 1. epochjának evaluation adathalmazon számolt metrikái:

<i>Hiperparaméter-hangolás tesztesetének neve (lr=learning rate, bs=batch size)</i>	<i>evaluation loss</i>	<i>precizitás</i>	<i>szenzitivitás</i>	<i>F1- mérték</i>	<i>pontosság</i>	<i>kiegyensúlyozott pontosság</i>
1e-5lr16bs-eng_x1-cased_L-12_H-768_A-12	0,1810	0,9494	0,9659	0,9576	0,9307	0,8734
1e-5lr32bs-eng_x1-cased_L-12_H-768_A-12	0,1836	0,9337	0,9799	0,9563	0,9275	0,8419
2e-5lr16bs-eng_x1-cased_L-12_H-768_A-12	0,1931	0,9386	0,9733	0,9557	0,9269	0,8512
2e-5lr32bs-eng_x1-cased_L-12_H-768_A-12	0,1694	0,9487	0,9686	0,9586	0,9322	0,8729
3e-5lr16bs-eng_x1-cased_L-12_H-768_A-12	0,1922	0,9435	0,9686	0,9559	0,9276	0,8609
3e-5lr32bs-eng_x1-cased_L-12_H-768_A-12	0,1718	0,9443	0,9728	0,9584	0,9316	0,8644
5e-5lr16bs-eng_x1-cased_L-12_H-768_A-12	0,2316	0,9373	0,9675	0,9522	0,9213	0,8460
5e-5lr32bs-eng_x1-cased_L-12_H-768_A-12	0,1811	0,9432	0,9714	0,9571	0,9295	0,8612
6e-5lr16bs-eng_x1-cased_L-12_H-768_A-12	0,2499	0,9318	0,9586	0,9450	0,9097	0,8299
6e-5lr32bs-eng_x1-cased_L-12_H-768_A-12	0,2090	0,9245	0,9797	0,9513	0,9188	0,8195
1e-5lr16bs-eng_x1-uncased_L-12_H-768_A-12	0,1777	0,9596	0,9574	0,9585	0,9329	0,8930
1e-5lr32bs-eng_x1-uncased_L-12_H-768_A-12	0,1703	0,9483	0,9725	0,9602	0,9348	0,8733
2e-5lr16bs-eng_x1-uncased_L-12_H-768_A-12	0,1816	0,9510	0,9674	0,9591	0,9332	0,8775
2e-5lr32bs-eng_x1-uncased_L-12_H-768_A-12	0,1644	0,9446	0,9776	0,9608	0,9355	0,8668
3e-5lr16bs-eng_x1-uncased_L-12_H-768_A-12	0,1839	0,9436	0,9716	0,9574	0,9300	0,8623
3e-5lr32bs-eng_x1-uncased_L-12_H-768_A-12	0,1623	0,9499	0,9712	0,9604	0,9352	0,8767
5e-5lr16bs-eng_x1-uncased_L-12_H-768_A-12	0,2036	0,9255	0,9791	0,9516	0,9193	0,8219
5e-5lr32bs-eng_x1-uncased_L-12_H-768_A-12	0,1799	0,9497	0,9646	0,9571	0,9300	0,8737
6e-5lr16bs-eng_x1-uncased_L-12_H-768_A-12	0,2115	0,9296	0,9739	0,9512	0,9192	0,8300

6e-5lr32bs-eng_x1-uncased_L-12_H-768_A-12	0,1812	0,9441	0,9716	0,9577	0,9305	0,8634
1e-5lr16bs-hun-hubert	0,1632	0,9676	0,9768	0,9722	0,9519	0,8879
1e-5lr32bs-hun-hubert	0,1338	0,9754	0,9709	0,9731	0,9539	0,9101
2e-5lr16bs-hun-hubert	0,1625	0,9568	0,9816	0,9690	0,9461	0,8546
2e-5lr32bs-hun-hubert	0,1495	0,9655	0,9767	0,9710	0,9499	0,8810
3e-5lr16bs-hun-hubert	0,1692	0,9720	0,9634	0,9676	0,9446	0,8963
3e-5lr32bs-hun-hubert	0,1443	0,9643	0,9755	0,9699	0,9479	0,8769
5e-5lr16bs-hun-hubert	0,2230	0,9470	0,9810	0,9637	0,9364	0,8216
5e-5lr32bs-hun-hubert	0,1755	0,9574	0,9760	0,9666	0,9420	0,8545
6e-5lr16bs-hun-hubert	0,3458	0,8920	0,9948	0,9406	0,8920	0,6269
6e-5lr32bs-hun-hubert	0,1796	0,9531	0,9759	0,9644	0,9381	0,8405
1e-5lr16bs-hun-hubert_wiki_lower	0,1687	0,9611	0,9730	0,9670	0,9429	0,8655
1e-5lr32bs-hun-hubert_wiki_lower	0,1539	0,9689	0,9619	0,9654	0,9407	0,8862
2e-5lr16bs-hun-hubert_wiki_lower	0,1985	0,9608	0,9740	0,9674	0,9436	0,8650
2e-5lr32bs-hun-hubert_wiki_lower	0,1614	0,9730	0,9564	0,9646	0,9397	0,8968
3e-5lr16bs-hun-hubert_wiki_lower	0,1874	0,9599	0,9734	0,9666	0,9422	0,8618
3e-5lr32bs-hun-hubert_wiki_lower	0,1567	0,9686	0,9642	0,9664	0,9423	0,8860
5e-5lr16bs-hun-hubert_wiki_lower	0,1697	0,9563	0,9744	0,9653	0,9397	0,8503
5e-5lr32bs-hun-hubert_wiki_lower	0,1582	0,9643	0,9681	0,9662	0,9417	0,8738
6e-5lr16bs-hun-hubert_wiki_lower	0,2662	0,9608	0,9108	0,9351	0,8913	0,8411
6e-5lr32bs-hun-hubert_wiki_lower	0,1616	0,9602	0,9734	0,9668	0,9425	0,8628