TDK-dolgozat

Kiss Mihály

Szegedi Tudományegyetem
Természettudományi és Informatikai Kar

Informatikai Intézet

Számítógépes Algoritmusok és Mesterséges Intelligencia Tanszék

Nagy nyelvi modellek által generált szövegek felismerésének többosztályos modellezése

Készítette:
Kiss Mihály
Programtervező
informatikus MSc

Témavezetõ:
Berend Gábor
docens

Szeged
2024

Tartalomjegyzék

Absztrakt

Manapság rengeteg mesterséges intelligencia (MI) által generált tartalom keletkezik. Ezt még az is elősegíti, hogy az átlagos, technológiához nem feltétlen értő emberek számára is könnyen elérhetővé váltak különböző modellek, mint például a ChatGPT, Bard, vagy Gemini. Ebből adódóan egyre nagyobb szükség van olyan algoritmusokra és megoldásokra, amelyek képesek az ember és gép által generált tartalmak megkülönböztetésére.

Ennek a problémának a megoldásán dolgoztam, amely egyben egy verseny is volt. A versenyhez járt egy adathalmaz, ami több mint 600.000 tanítóadatnyi szöveggel rendelkezett, amelyet vagy ember írt, vagy pedig 40 különböző nagy nyelvi modell egyike. A végső előrejelzést bináris formában kellett megadni arra vonatkozóan, hogy az adott szöveget ember vagy MI hozta létre. A gépileg generált szövegek azonban nem homogének, – hiszen a szöveggenerálásra használt modellek nagyon különbözőek: jelentős eltérések vannak a paraméterszámukban és az általuk generált szövegek minőségében. Ennek fényében dolgozatomban azzal a feltételezéssel éltem, hogy a feladatot — noha az eredetileg bináris problémaként került megfogalmazásra -– érdemes lehet nem bináris megközelítésben kezelni a tanítás során, és előnyös, ha a modelleket bizonyos tulajdonságaik és teljesítményük alapján különböző csoportokra osztjuk. Tehát a tanítás több osztályra történt, amely kiértékelésnél redukálva lett binárisra az alapján, hogy emberi szövegnek vagy valamely nyelvi modell által generáltnak jelezte. Emellett vizsgáltam még a szövegek hosszalapú megkülönböztetését is, ahol szöveghossz alapján külön-külön voltak betanítva a modellek. A legtöbb kísérletre alkalmaztam egy szavazást is, ahol 3 modell szavazott az adott szövegről, és ennek vettem az átlagolt súlyozott valószínűségét minden osztályra a végső döntés meghozatala során. A modellek szavaztatásának költségeinek csökkentése érdekében — ismereteim szerint elsőként — teszteltem a korábban konvolúciós hálókra kifejlesztett ún. snapshot ensemble eljárást.

A többosztályos megoldások közül legtöbb is jobb eredményt hozott, mint a bináris tanítás. A legkiemelkedőbb teljesítményt a teszt halmazra az az osztályozás adta, amelynél minden modell egy osztályt képviselt. Ez az eredmény szavazási módszerrel még jobb lett, így ortogonális javulást hozott a többosztályos megoldás és a szavazás együttes használata. A versenyen 36 indulóból a 7. helyen végzett az általam beadott megoldás.

1. fejezet

Bevezetés

Az elmúlt években a mesterséges intelligencia (MI) gyors fejlődése lehetővé tette, hogy egyre jobb eredményeket érjenek el alapvető automatizált feladatok terén. Ezzel párhuzamosan az MI képességei egyre szélesebb körben ismertté váltak, kezdve a képgenerálástól egészen a szöveges tartalmak generálásáig. Különösen nagy figyelmet kapott a természetes nyelvfeldolgozás területe, ahol a state-of-the-art nyelvi modellek képesek rendkívül összetett szövegek előállítására, amelyek hasznosak lehetnek, de gyakran már nehezen megkülönböztethetők az emberi írásoktól.

Ezeket a tartalmakat a nagy nyelvi modellekkel (LLM) készítik, amelyek olyan algoritmusok, amelyeket hatalmas mennyiségű szöveges adaton hoznak létre. Az LLM-ek működése a neurális hálózatokon alapul, amelyek mesterséges ideghálózatok mintájára épülnek fel. A neurális hálózatok a bemeneti adatokból (szöveg, kép, hang) tanulnak, és fokozatosan felismerik a nyelvi mintákat és összefüggéseket. A tanulás során a nagy nyelvi modellek hatalmas szókincset építenek ki, és megtanulják a nyelvtani szabályokat, a stílusbéli sajátosságokat és a szövegkörnyezetet is. Például az OpenAI GPT-4 [14] az egyik legnépszerűbb nagy nyelvi modell. Egyes források szerint körülbelül 1.8 trillió gépi tanulási paraméterrel rendelkezik [6] és jó képességgel generál szöveget. Leveleket fogalmaz, cikkeket és verseket ír, nyelvek között fordít. Egy másik figyelemre méltó példa a Google által fejlesztett Gemini [4], ami a GPT-4-hez hasonlóan egy hatalmas paraméterszámmal rendelkező modell.

Manapság rengeteg MI által generált tartalom keletkezik. Ezt még az is elősegíti, hogy az átlagos, technológiához nem feltétlen értő emberek számára is könnyen elérhetővé váltak különböző modellek. Ebből adódóan egyre nagyobb szükség lesz olyan algoritmusokra, amelyek képesek az ember és gép által generált tartalmak megkülönböztetésére. A nagy nyelvi modellek fejlődésének gyorsasága és a gépi szövegek elterjedése számos új kérdést vet fel, többek között a szerzői jogok, a hamis információk és a biztonság területén is.

A nagy innovációs fejlődés meglátszik a piaci viszonyokon is. A mesterséges intelligencia piaca 2023-ban elérte a 207,9 milliárd dollárt, és 2030-ra várhatóan 1,85 billió dollárra fog emelkedni [19]. Ez kilencszeres növekedést jelent, de ennél akár nagyobb is lehet. Jól mutatja a hatalmas piaci fejlődést, hogy 2021-ben még kevesebb mint 100 milliárd dollárt ért ez az ágazat. Az MI piacának másik részét képezik a mesterséges intelligencia által készített tartalmakat detektáló eszközök, amelyek fontos szerepet fognak

játszani a biztonság, az adathalászat elleni védekezés, a hamis információk azonosítása és a csalások megelőzése területén. Ezen eszközök piaca is gyors növekedést mutat, amit a kiberbiztonsági fenyegetések növekvő száma és a digitális adatok mennyiségének robbanásszerű növekedése táplál. Pontos adatokat nem lehet találni arról, hogy mekkora erőforrásokat tesznek a mesterséges intelligencia ágazatába, de arról lehet információkat találni, hogy az OpenAI – a generatív modellek nagy ismeretségét meghozó, ChatGPT-t is létrehozó cég – is vizsgálta ennek lehetőségeit. Ki is adtak egy modellt még 2023 januárjában, de júliusra már meg is szüntették az elérhetőségét, aminek a hátterében az állt, hogy a modell minősége nem volt kielégítő: mindösszesen nagyjából 26%-os pontossággal volt képest meghatározni azt, hogy egy szöveget ember vagy mesterséges intelligencia hozott-e létre [13]. Amellett, hogy megszűntették, információkat is közöltek arról, hogy mik segíthetik elő a minél pontosabb felismerését az MI által generált szövegeknek. Ezek között volt az, hogy minél hosszabb egy szöveg, annál nagyobb az esély a helyes döntés meghozatalára, valamint hogy a szöveg stílusa, témája is képes befolyásolni a felismerés hatékonyságát, mivel a neurális hálózatokra épülő osztályozók nem működnek jól, ha a tanítási adatoktól eltérő eloszlásból jövő adatot kapnak. A probléma összetettségét és nehézségét jól mutatja, hogy még a piacvezető vállalat szakemberei is azt mondják, hogy nehéz a doménagnosztikus szövegek eredetének felismerése.

Ebben a dolgozatban a COLING2025 AI-generált tartalmak felismerésére irányuló versenyén [21] elért megoldásom [8] eredményeit mutatom be. A megoldások arra a részfeladatra készültek, amelynek célja egy bináris előrejelzés készítése volt, aminél azt kell megmondani, hogy az adott szöveg emberi vagy nagy nyelvi modell által generált. A feladat adathalmaza kizárólag angol nyelvű szövegeket tartalmazott. E kihívás kezelésére különféle osztályozási sémákkal kísérleteztem annak érdekében, hogy megtaláljam a gép által generált szövegek felismeréséhez megfelelő módszert. Bináris, 3, 5, 6 és 41 osztályos megoldásokat fejlesztettem ki, ahol minden konfiguráció a modellek paraméterei, mérete vagy a generált szöveg minősége alapján különböző heurisztikus csoportosításokat képviselt. A robusztusság és a teljesítmény javítása érdekében a legtöbb osztálykonfigurációhoz szavaztatás alapú standard ensemble módszereket alkalmaztam. További kísérletként egy snapshot ensemble-t [7] is létrehoztam. Továbbá kifejlesztettem egy hosszalapú megoldást is, amelyben a szövegek hosszuk alapján lettek kategorizálva. Három intervallumot hoztam létre a szöveg hossza alapján, és mindegyik intervallumhoz egy-egy szakértő modell lett finomhangolva. Ezáltal minden szöveg egy szakértő modellhez került, remélve, hogy ennek a módszernek a használata javulást eredményez a pontosságban.

Végső beküldésem a 6-osztályos megoldás volt, 0,791-es makró F1-pontszámot ért el, bár nem ez volt a legjobban teljesítő modell. A verseny utáni értékelés kimutatta, hogy a hosszalapú megközelítés 0,825-ös makró F1-pontszámot ért el, míg a 41 osztályos megoldás szavaztatással kicsivel magasabb, 0,826-os pontszámot produkált. Az egyes kísérletek forráskódjai elérhetőek a Szeged_MGT GitHub repozitóriumában [1].

[1] https://github.com/kissmihaly1/Szeged_MGT

2. fejezet

Kapcsolódó kutatások

Ez a fejezet a releváns kutatásokat tekinti át, amelyek az LLM-ek által generált szövegek észlelésével és hatásaival foglalkoznak.

Az élet számos területén megjelentek már a szintetikus szövegek, ami alól a tudományos szövegek sem jelentenek kivételt. Egy korábbi cikkben [10] a kutatók egy megoldást mutattak be, amely képes becsülni, hogy egy nagy szövegkorpuszon belül mekkora az LLM-ek által lényegesen módosított vagy generált szövegek aránya. Ez a módszer szakértői szövegek és mesterséges intelligencia által generált tartalmak összehasonlításával elemzi a nagy nyelvi modellek valós felhasználását. Az eredmények szerint a benyújtott bírálatok 6,5%-16,9%-a lehetett jelentősen LLM-ek által módosított. Különösen a határidő közelében beküldött, alacsonyabb magabiztosságot jelző és a szerzői válaszokra kevésbé reagáló bírálatok esetében volt magasabb az LLM-ek használatának valószínűsége. Egy másik elemzés [9] 950,965 tudományos cikket vizsgált meg, amelyek 2020 januárja és 2024 februárja között jelentek meg különböző archívumokban. A kutatók egy populációszintű statisztikai keretrendszert használtak az LLM-ek által módosított tartalom előfordulásának mérésére anélkül, hogy egyedi eseteken kellett volna következtetéseket levonniuk. Az eredmények azt mutatták, hogy az LLM-ek használata folyamatosan növekszik, különösen a számítástudományi témájú cikkeknél, ahol az arány akár 17,5%-ot is elérhet. Ezzel szemben a matematikai cikkeknél ez az arány jelentősen alacsonyabb volt (legfeljebb 6,3%). Az elemzés szerint magasabb LLM-módosítási szintek jellemzőek voltak olyan publikációkra, ahol az első szerzők gyakrabban posztolnak preprinteket, népszerűbb tudományterületeken jelennek meg, vagy rövidebb terjedelműek. Ezek az eredmények arra utalnak, hogy az LLM-ek széles körben elterjedtek a tudományos írásban, és jelentős hatással vannak a tudományos közösség gyakorlatára.

Az utóbbi időben több verseny is volt a szintetikus szövegek felismerésére, azokból is lehet tanulságokat levonni, illetve megfigyelni az ott készült megoldásokat. Az egyik ilyen nagyobb verseny, az a 2024-ben rendezett SemEval Task 8 [22] volt. Ezen a versenyen több részfeladat is volt, két fontosabb ezek közül: bináris, és többosztályos felismerés. A többosztályos itt azt jelenti, hogy a felismerés során azt kellett megmondani, hogy ember írta, vagy esetleg modell generálta a szöveget. A bináris verseny győztesének megközelítése [16] egy olyan transzformer enkódert alkalmaz, amely több LLaMA-2 modellből származó valószínűségi jellemzőket kombinál minden tokenre. Ez a modell a legjobb pontosságot érte el a versenyen. A többosztályos részfeladaton a győztes a szavaztatás

módszerét alkalmazta, ami azt jelenti, hogy több különálló modellt finomhangoltak, és ezen modellek között szavaztatást alkalmaztak. Ennek segítségével egy robusztusabb és pontosabb kiértékelést értek el.

A standard ensemble módszer már volt korábban is használva erre a feladatra, mégpedig az Automated Text Identification (AuTexTification) [17] versenynél. A verseny keretében két fő feladatot volt: (1) annak meghatározása, hogy egy adott szövegrészlet MI által generált-e vagy emberi eredetű, és (2) a szöveget generáló konkrét nyelvi modell. Ezen a versenyen az elemzések angol és spanyol nyelvű szövegeken alapultak. Az első feladatban a csapat modellje angol szövegeken az ötödik, spanyol szövegeken pedig a tizenharmadik helyezést érte el. A második feladatban, a modellel angol és spanyol nyelven is első helyezést tudtak elérni a szerzők, ami jól illusztrálja a modellek szavaztatásának hatékonyságát. Ez ösztönzött arra engem is, hogy használjam a többosztályos megoldásoknál.

Készült korábban egy tanulmány [1], amely szintén a mesterséges intelligencia által generált szövegek felismerhetőségével foglalkozik, azon belül is leginkább a jelenlegi felismerő módszerek hiányosságait tárja fel. Ehhez javasoltak egy rangsort, amely Counter Turing Test (CT2) névre lett keresztelve, amelynek a célja, hogy átfogó értékelést nyújtson a meglévő technikák robusztusságáról. Emellett még egy metrikát is létrehoztak, amelynek a neve „AI Detectability Index” (ADI). Jelenleg 15 jónak számító nagy nyelvi modellt osztályoztak ezzel a metrikával, és a nagyobb nyelvi modellek nagyobb értéket értek el a kisebb méretű nyelvi modellekkel szemben, amely azt jelzi, hogy kevésbé észlelhető, hogy mesterséges intelligencia által készültek. A tanulmány fő célja, hogy elősegítsék az észlelhetőséget, és hogy akár szakpolitikai döntéshozatalnak is része legyen az ő általuk publikált megoldás.

3. fejezet

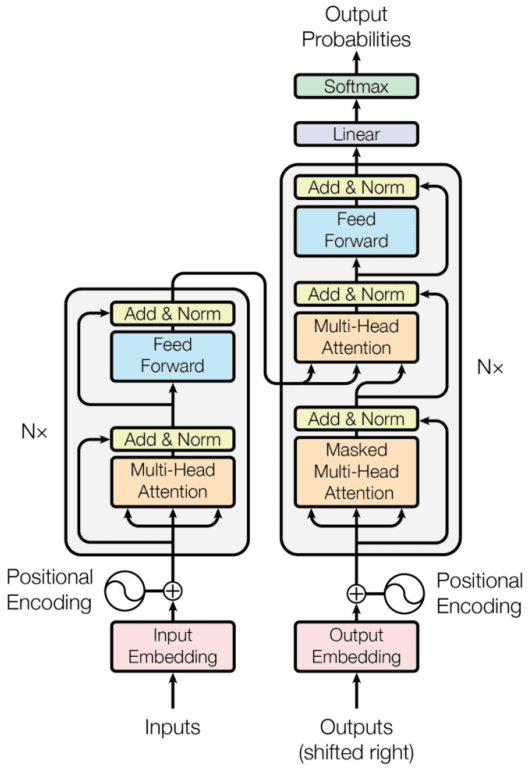
Felhasznált technológiák

3.1. Transzformer architektúra

A transzformer [20] architektúra az egyik legjelentősebb előrelépés volt a természetesnyelv-feldolgozásban, és úgy általánosságban a gépi tanulásban.

Noha számos kihívóra akadt – pl. az állapottérmodellek (state space models) formájában –, a mai napig is az egyik legjobb gépi tanulási megoldásnak számít.

A modell működésének megértése érdekében ismernünk kell a szóbeágyazásokat. A nyers szöveges adatot, amivel rendelkezünk, nem lehet közvetlenül használni a transzformer modellek tanításához, azokat először vektoros formára kell alakítani. Vagyis ahelyett, hogy magukat a tokeneket továbbítanánk a modellben található enkódernek, az egyes tokenekhez tanult vektoros reprezentációkat, az ún. beágyazásokat továbbítjuk. Mindez még nem kódolja a tokenek mondaton belüli elhelyezkedését. A tokenek pozicionális információjának kódolásához egy másik beágyazást, az úgynevezett pozícióbeágyazást lehet használni. Itt nem a szavak tokenjeit adjuk meg bemenetként, hanem azok pozíció-azonosítóit vagy indexeit, ami lehetővé teszi, hogy az egyes pozícióknak hasznos reprezentációit tanulja meg a modell. A beágyazási réteg végső kimenete az a tokenbeágyazás és a pozícióbeágyazás összeadásából, és az így kapott eredmény normalizálásából adódik.



3.1. ábra. Transzformer felépítése Forrás: [20]

A 3.1 ábrán látható transzformer architektúra megértése 3 fő komponenssel lehetséges:

1. **Ön-figyelmi mechanizmus (self-attention)**: lehetővé teszi a modellünknek, hogy a szekvenciális input adatok megfelelő aspektusaira fókuszáljon. Ez többször is használatos mind az enkóderben, mind a dekóderben (amennyiben a modell rendelkezik vele), hogy segítsen nekik összefüggésbe helyezni a bemeneti adatokat.

2. **Enkóder (encoder)**: kódolja a beérkező sorozatokat reprezentációs vektorokká alakítja.

3. **Dekóder (decoder)**: reprezentációs vektort dekódolja, hogy ezáltal legenerálja a célsorozatot. A TDK dolgozatban létrehozott modellek ilyen dekódoló komponenssel nem rendelkeztek.

Az ön-figyelmi mechanizmus elnevezés onnan ered, hogy az „általános” figyelemmechanizmussal ellentétében az ön-figyelem ugyanarra a szekvenciára utal, amelyet éppen kódol.

Az önmagára figyelő réteg képes figyelembe venni a szavak vagy tokenek kontextusát, azok egymáshoz való viszonyát egy mondatban vagy mondatok halmazában, és ezeket az információkat is kódolja az egyes tokenekhez létrehozott beágyazásaiban.

A kontextus beágyazásba való kódolása fontos feladat, mert egy szöveg homonimái ugyanahhoz a beágyazáshoz lesznek rendelve. Vegyük például azt a szöveget, amely tartalmazza a *„vár”* kifejezést. A *„vár”* egy szövegben használható, mint egy ember, aki

várakozik, vagy abban az értelemben is, mint egy erőd, amit katonák védenek. A legelső enkóder réteg inputjaként funkcionáló beágyazásoknak ugyanaz lesz az beágyazási értékük mindkét értelmezés esetén (amennyiben a kérdéses szavak az input ugyanazon tokenpozícióján helyezkednek el), de ezt nem szeretnénk. Amikor az önmagára figyelő réteg befejezte az átadott beágyazásokkal való munkát, az adott réteg outputjaként előálló beágyazásokban kódolva lesz a kontextusuk. Például a *„vár”* szónak az önfigyelmi mechanizmust használó enkóder rétegek kimeneteiként előálló rejtett vektorai jobban fognak hasonlítani az erőd (építményi értelemben használt) szó vektoros reprezentációira, ha egy szövegben úgy használták, hogy „III. Béla uralkodó a Budai *vár* felújítását rendelte el”.

Az ön-figyelem a kontextust három tenzor - Query (Lekérdezés), Key (Kulcs) és Value (Érték) - használatával és ezekhez a tenzorokhoz kapcsolódó matematikai műveletek definiálják. A tenzorok egy tanulható lineáris leképezések segítségével jönnek létre az adott réteghez továbbított rejtett vektorokból. Ezeknek a tenzoroknak a jelentése egy egyszerű példával egészen érthetővé válik. Amikor "Végtelen háború"-t keresünk a Google-ben, a keresőszöveget, amit beírunk, azt lekérdezésnek hívják. A Google szerverei hasonló lekérdezésekhez hash kulcsokat tartanak nyilván, és ezeket a kulcsokat összehasonlítják a lekérdezéssel, hogy megtalálják a hasonlósági pontszámokat. Ezek a hash kulcsok a fentebb tárgyalt kulcsok. Végül a felhasználónak visszaküldött tartalom az érték. Az önmagára figyelő mechanizmusban a lekérdezés az enkóderből származik. A lekérdezés és kulcs tenzorok közötti mátrixszorzás adja meg egy token (szó) hasonlósági pontszámát az összes többi tokennel. Mivel a mátrixszorzásból származó értékek nagyok lehetnek, szoftmax függvényt alkalmazunk a pontszámmátrixon, majd ezt szorozzuk az értékmátrixszal az egyes rétegek kimeneteinek az előállításának az érdekében. Ennek a képlete:

$$\text{Kimenet}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$$

ahol Q a lekérdezési mátrix, K a kulcsmátrix, V az értékmátrix, d_k pedig a K mátrix dimenziójának mérete.

A következőkben bemutatom a „Multi-Head”, azaz többfejű figyelmi mechanizmust. Alapvetően a többfejű figyelem több, párhuzamos ön-figyelem egységből áll. Ez azért hasznos, mert így minden figyelem egységnek lehetősége van arra, hogy egy adott szövegjellemzőre összpontosítson. Egy figyelem egység például a tárgy-igekapcsolatra, míg a másik a szöveg időtartamára fókuszálhat, és így tovább. Ez hasonló például a több szűrő használatához egyetlen konvolúciós rétegben. A több modell általában jobb eredményekhez vezet. Általában a többfejű figyelmi réteg az utolsó dimenziót (beágyazási dimenziót) egyenlő részekre osztja a figyelmi egységek skálázhatósága érdekében, és az egyes figyelmi egységek kimeneteit összefűzi, majd ezekre az eredményekre lineáris transzformációt alkalmaz a végső kimenet meghatározása érdekében. A lineáris transzformációt azért alkalmazzák, hogy a generált kimenet hasznosabb legyen a később átadott úgynevezett előrecsatolt neurális háló rétegre. Ez a réteg az egyik legismertebb fajtája a neurális hálózatoknak, amelynek az a lényege, hogy az információ csak egy irányba áramolhat, általában a bemeneti rétegtől, a rejtett rétegeken át, a kimeneti rétegig átvezető utat jelenti. Ez a hálózat általában két lineáris rétegből áll, köztük egy ReLU aktivációs függvénnyel. Ez arra jó, hogy a modell mélyebb jelentést nyerjen ki a bemeneti adatokból, így tömörebben és hasznosabban ábrázolja a bemenetet. A transzformer enkóder architektúrája jellemzően

több rétegből áll, amely rétegek mindegyike tartalmaz egy ön-figyelmi mechanizmust és egy előrecsatolt neurális hálózatot, valamint reziduális kapcsolatokat, amelyek segítik az információk megőrzését a rétegek között.

Az enkóder architektúrát önmagában leginkább szövegek osztályozására használják, ahol a modellnek osztályoznia kell a szövegeket már előre definiált osztályokba. Ilyen például a spam detektálás, vagy szemantikai elemzés alapján kategóriákba sorolás. Az enkóder rétegnek tokenek egy sorozatát kell megadni, és az visszaad egy fix hosszal rendelkező vektoros reprezentációt az egész sorozatról, amelyet utána fel tud használni az osztályozás végrehajtásához. Az ön-figyelem mechanizmus lehetővé teszi, hogy a modell mérlegelje a különböző bemeneti szekvenciarészek fontosságát.

Tehát az elejéről tekintve a teljes folyamat: adatok először tokenizálásra kerülnek, majd minden tokenhez hozzáadódik egy pozíciós kódolás, amely megadja a tokenek sorrendjét és segít a modellnek a szöveg kontextusának megértésében. Az adatok ezután több önfigyelmi rétegen mennek keresztül, ahol a modell kiértékeli a tokenek közötti kapcsolatokat. Minden token számára a modell meghatározza, hogy mely más tokenek fontosak a jelentés szempontjából. Az önfigyelmi réteget követően egy reziduális kapcsolat van, valamint egy normalizáló lépést is végrehajt a tanulási folyamat stabilizálása érdekében. Ezt követi egy előrecsatolt neurális hálózat, ami további feldolgozást végez, finomítva az információkat. Az enkóder végén az adatokat ismét egy reziduális kapcsolat és normalizáló lépés összegzi, ami után a teljes enkóder kimenete egy mély, kontextusban gazdag reprezentációt nyújt a bemeneti adatokról. Összegezve a transzformer architektúra enkóder része egy olyan mélytanulási megoldás, amely képes az összetett kapcsolatok azonosítására az adatokban a figyelem mechanizmus segítségével. Ezzel lehetővé teszi a modell számára, hogy dinamikusan súlyozza a bemeneti adatok fontosságát, így jó eredményeket ér el különböző nyelvi feladatokon. A dolgozatban alapvetően a transzformer modellek közül olyan „encoder-only” modelleket használtam, amik kizárólag az enkóder komponensét használják az eredetileg javasolt transzformer architektúrának.

3.2. BERT, DeBERTa

Mindkettő modell a transzformer architektúrára alapszik, de mégis különböző megközelítésekkel javítják az eredményeiket a nyelvi feladatokon.

A BERT (Bidirectional Encoder Representations from Transformers) [2] modellt 2018-ban adták ki, szintén a Google kutatói, csakúgy, mint a transzformer architektúránál. Ennek a modellnek a nagy újdonsága az volt, hogy mindkét irányba párhuzamosan tud „olvasni”. Ez a korábbi módszereknél nem volt elérhető. A transzformer architektúra segítségével a modell minden szóhoz hozzáfér mind a bal, mind a jobb oldali kontextushoz a tanulási folyamat minden egyes lépésében, amit "teljesen irányítatlan" módszernek is nevezünk. Ez azt jelenti, hogy amikor a BERT egy adott szót próbál értelmezni vagy a szó jelentését beágyazni, nem csak az előtte lévő szavakból merít információt, hanem az utána következőkből is. Például az "A kutya ugrott a kerítés felett" mondatban, amikor a BERT a "kerítés" szót elemzi, nem csak a "A kutya ugrott a" szavak alapján dolgozik, hanem figyelembe veszi a "felett" szót is. Ezzel a módszerrel a modell képes a szöveg mélyebb, kontextusfüggő értelmezésére és pontosabb jelentéstanulásra, mivel a szavak

jelentése gyakran függ mind a bal, mind pedig a jobb oldali környezetüktől. Dolgozatomban a „cased” verzióját használtam a BERT-nek, amely abban különbözik a szavak kapitalizációjára érzéketlen („uncased”) verziótól, hogy megkülönbözteti a nagy-, és kisbetűket is.

A DeBERTa (Decoding-enhanced BERT with Disentangled Attention) [5] modell már a Microsoft által lett publikálva. A fő újítása az volt, hogy a modell a figyelemmechanizmusban megkülönbözteti a pozíciós és a tartalmi kapcsolatokat. Ez a megkülönböztetett figyelem lehetővé teszi a modell számára, hogy pontosabban értelmezze a szövegkörnyezetet, ami fokozza a teljesítményét különféle feladatokban. A DeBERTa egy másik fontos összetevője az Enhanced Mask Decoder (EMD), amely a maszkolt nyelvi modellezési feladatot javítja. Az EMD egy újfajta dekóder réteget alkalmaz, amely kifejezetten a maszkolt szavak helyreállítására összpontosít. Ez a réteg segít a modellnek abban, hogy jobban megértse és pontosabban rekonstruálja a maszkolt szavakat a kontextus alapján. Azt hozzá kell tenni, hogy ez a modell jóval lassabb az azonos méretű BERT modell használatához képest.

3.3. Ensemble Módszerek

3.3.1. Standard Ensemble

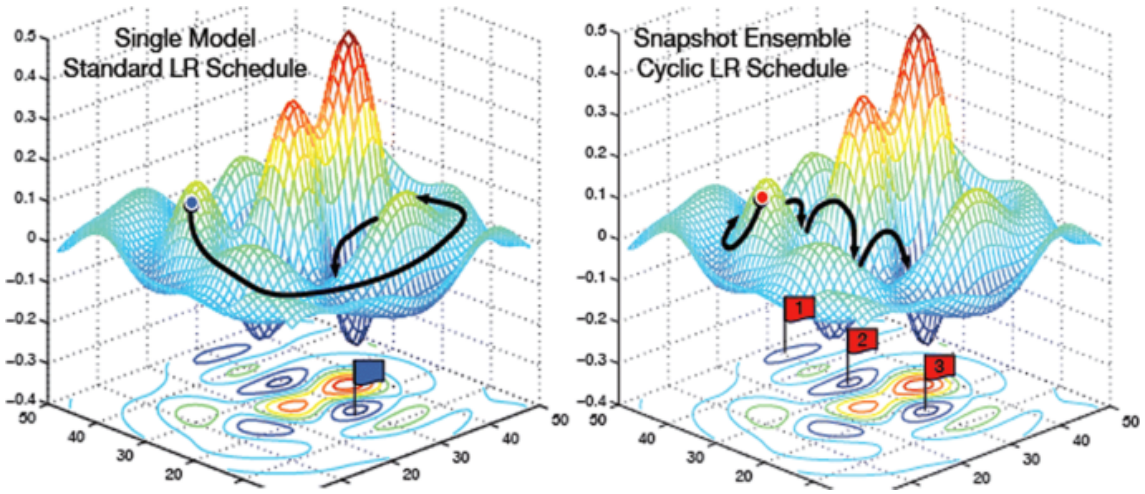
A standard ensemble (másnéven szavaztatás) egy olyan módszer a gépi tanulásban, ahol több, egymástól független modell együttesen járul hozzá a végső előrejelzéshez, növelve ezzel a predikciók pontosságát és megbízhatóságát [23]. Ebben a megközelítésben különböző algoritmusok vagy azonos típusú modellek, például döntési fák, neurális hálózatok vagy nagy nyelvi modellek kombinálhatók. Az ensemble technikákkal, például többségi szavazással, átlagolással vagy súlyozott eredményekkel biztosítja, hogy az egyes modellek által produkált hibák kiegyenlítsék egymást, ezáltal javítva a rendszer teljesítményét. Ezzel a rendszer egy jobb általánosítóképességet kaphat. A javulás ellenére a szavaztatási módszerek számításigényesek, és mivel több modellt kell betanítani és használni, így idő- és erőforrásigényük jelentős lehet.

3.3.2. Snapshot Ensemble

Dolgozatomban az ún. snapshot ensemble megoldást is megvalósítottam, amely során a modell több „pillanatképét” mentettem a tanulási folyamat különböző szakaszaiban. Ellentétben a standard ensemble-el, amelyek több, független modellt betanítását hajtják végre az alapoktól, ez a módszer ugyanannak a modellnek a változó állapotait használja fel újra. Azáltal, hogy ezeket a köztes „pillanatképeket” alkalmazzuk, a módszer változatosságot visz az előrejelzésekbe anélkül, hogy külön modellekre lenne szükség, ezáltal időt és számítási kapacitást takarítva meg [7]. Ez egy kiegyensúlyozott megoldást nyújt, amely javíthatja a pontosságot, miközben csökkenti a hagyományos ensemble módszerekre jellemző erőforrásigényt.

Mindezt úgy valósítja meg, hogy a tanítás során a tanulási rátát periodikusan csökkenti és növeli. Ez a megközelítés segít a modellnek különböző lokális minimumokba kon-

vergálni, így az egyes „pillanatképek” eltérőek, és változatos predikciókat biztosítanak. Legjobb tudomásom szerint elsőként vizsgáltam a snapshot ensemble eljárás hatékonyságát transzformer architektúrájú modellek esetén. A 3.2 ábrán a sztochasztikus gradiens csökkenés módszerével történő tanítás időbeli lefutásának sematikus ábrája látható a tanulási ráta tipikus csökkentése mellett (bal oldal), illetve a snapshot ensemble módszer által előírt tanulási ráta módosítási stratégia használatával (jobb oldal).



3.2. ábra. Snapshot ensemble működésének illusztrációja. (Ábra forrása: [7])

4. fejezet

Adatbázis és a feladat

4.1. Adatbázis

A verseny [21], amelyen részt vettem, a korábban is említett SemEval versenykiírásnak [22] a folytatásaként tekinthető. A verseny szervezői által biztosított adathalmaznak két fő része volt, az egyik a tanító, a másik a validációs halmaz. Az előbbiben körülbelül 611.000 szöveg volt megtalálható, az utóbbiban 262.000. Emellett közre adtak egy tesztelési halmazt is a fejlesztési fázis alatt, nagyjából 30.000 adatsorral. Ez viszonylag sok adat a korábbi hasonló versenyekhez képest, így remélni lehetett, hogy a modellek jól fognak teljesíteni. Az adathalmaz tartalmazott több jellemzőt minden egyedre, ezek között szerepelt az, hogy melyik modell készítette, ha nem emberi a szöveg. Emellett természetesen ott volt a szöveg, illetve az is meg volt adva, hogy melyik doménből érkezett az adott szöveg. A szövegek forrása diverznek mondható, megtalálhatóak voltak közöttük egyebek mellett például Wikipedia-ról, redditről, arxiv-ról vagy a yelp-ről származó szövegek is. Amit még érdemes tudnunk az adatbázisról, hogy a modellek halmaza is elég nagy, összesen 40 modell által generált szöveget tartalmazott. Ezek között szerepelnek kisebb-nagyobb modellek, kezdve a `flan_t5` modellektől az újabb `GPT` modellekig. A modellek eloszlása nem volt egyenletes, a `GPT` modellek által generált szövegekből volt a legtöbb, míg egy-két kisebb modell csak néhány ezer generált szöveggel volt jelen az adathalmazban.

4.2. Feladat

A verseny több feladatot is kiírt, amelyek közül én csak az első feladatra koncentrálok a dolgozatomban, ami a bináris felismerése a nagy nyelvi modellek által generált szövegeknek. Ezen belül volt két részfeladat is, az egyikben kizárólag angol nyelvű szövegek fordultak elő, míg a másikban angoltól különböző nyelvű szövegeket is lehetett találni. Én az angol nyelvű feladat megoldására készítettem el megoldásaimat. A feladat megoldása alatt tilos volt a szervezők által megadott adathalmazon kívül bármilyen adatot felhasználni, így kísérleteim során én is ennek megfelelően jártam el.

5. fejezet

Kísérletek bemutatása

5.1. Alkalmazott modellek

A megoldások kivitelezéséhez választani kellett egy enkóder modellt, ezért összehasonlítást végeztem a BERT és DeBERTa között. A két modell jellemzői láthatóak a 5.1 táblázatban.

Modell neve	Paraméterszám	Rétegek száma	Rejtett réteg mérete	Verzió
BERT	110 millió	12	768	bert-base-cased
DeBERTa	138 millió	12	768	deberta-base

5.1. táblázat. Alkalmazott modellek jellemzői

Az értékek alapján sejthető, hogy a DeBERTa jobb eredményt fog elérni. Végeztem egy finomhangolást mindkét modellre az alap bináris feladatra, hogy le tudjam mérni mindkettő teljesítményét. A DeBERTa minden esetben felülmúlta a BERT-et ebben a bináris osztályozási feladatban, magasabb F1-pontszámokat elérve. A BERT 0,973 makró F1-pontszámot ért el a validációs adathalmazon, míg a DeBERTa 0,982-t. Az eredmények alapján az összes megoldást a DeBERTa modellel készítettem.

5.2. Többosztályos kísérletek

Miközben a fő feladat bináris osztályozási problémaként lett megfogalmazva, ez nem biztos, hogy teljes mértékben visszaadja a gép által generált tartalmak sokszínűségét és változatosságát. A gép által generált szövegek kategóriája ugyanis heterogén csoportot alkot; olyan kimeneteket foglal magába, amelyek számos modelltől származnak, különböző paraméterszámokkal, tanulási adatokkal és szöveg-generálási teljesítménnyel. Például ugyanarra a kérdésre a `GPT-4` valószínűleg összefüggőbb és kontextusban pontosabb szöveget állít elő, mint a `flan_t5_small`. A gépileg előállított szövegek felismerésének bináris osztályozásként való kezelése a generált szövegek heterogenitásából adódóan nemkívánatos „olvasztótégely hatást” eredményezhet, amely során elveszhetnek lényeges

jellemzői az adatoknak. Ezen kihívások kezelésére kezdetben egy többosztályos osztályozási megközelítést fogalmaztam meg. Fontos megjegyezni, hogy bár a modelleket több osztályra tanítottam, minden kísérletnél bináris osztályozássá redukálom a kimenetet. Ha a modell bármilyen más osztályt jósol, mint az „ember”, azt „1”-ként (generált) kezeljük; ellenkező esetben „0”-ként (ember) kerül besorolásra egy adatpont. A finomhangolási kísérletek elvégzése során kivétel nélkül az alábbi hiperparamétereket használtam:

- **Optimalizáló eljárás**: AdamW [12]
- **Tanulási ráta**: 2e-5
- **Súlycsökkentés** (weight decay): 0,01
- **Bemelegítési lépések** (warmup steps): a teljes lépések 10%-a
- **Köteg** (batch) méret: 16
- **Epoch**: 3, korai leállítással (early stopping)

A finomhangolást minden kísérlet alatt legalább 3, különböző véletlen seed alapján inicializált osztályozási súlyokkal rendelkező modellre elvégeztem, majd a kapott eredmények átlagoltam. Ez azért fontos, mert bár az előtanított DeBERTa modellek azonos kiinduló paraméterekkel rendelkeznek az enkóder tekintetében, a seed értékek befolyásolják a finomhangolási folyamat egyes véletlenszerű elemeit (pl. maguknak az osztályozás elvégzéséért felelős modellparamétereknek a kezdeti értékeit). Ennek eredményeképpen a különböző seed értékek használata mellett finomhangolt modellek (akár jelentősen is) eltérő eredményeket produkálhatnak. Az átlagolással csökkenthető a véletlenszerűség hatása, így pontosabb és robusztusabb teljesítményértékelést érhetünk el.

5.2.1. 41-osztály: modellenként egy osztály

Mivel az adathalmaz tartalmazta azt az információt, hogy melyik modell generálta a szöveget, célom volt ennek kihasználása, és minden modellt külön osztályként kezeltem, ami ebben az esetben 41 osztályt jelentett. Ez a módszer azonban túlzottan szélsőségesnek tűnhet, ami fragmentált adathalmazt eredményezhet, ahol az egyes osztályok korlátozott mennyiségű adatot tartalmazhatnak a robusztus tanításhoz. Emellett az is ronthat az eredményen, hogy olyan osztályt kellene prediktálni a teszthalmazban, ami a tanítóadatban nem volt jelen. Emiatt ez a megközelítés túlságosan egyszerűsítő lehet, érdemes lehet a modellezés során kevésbé részletes osztályokat megkülönböztetni egymástól.

Ennek érdekében további 3, 5 és 6 osztályos megoldásokat is definiáltam. Ezen eltérő osztályszerkezetek alkalmazásával célunk olyan kategóriák létrehozása volt, amelyek jól elkülönülő tulajdonságokkal rendelkeznek, így a modell hatékonyan tud különbséget tenni közöttük.

5.2.2. Osztályok definiálása paraméterszám alapján

5 osztály használata

Második megközelítésem során a modelleket a paraméterszámuk alapján csoportosítottam, amely kategorizálást a modellek által generált szövegek minősége és a modellek paraméterszáma közötti összefüggéssel kapcsolatos hipotézis motiválta. Ez a csoportosítási stratégia lehetőséget ad annak vizsgálatára, hogy a különböző paraméter-skálájú modellek eltérő viselkedést vagy szöveggenerálási képességeket mutatnak-e bináris osztályozási feladatokban. A modellek paraméterszám szerinti szegmentálásával arra törekedtem, hogy kiemelje az esetleges trendeket a modellek hatékonyságában, és jobban megértsem, hogyan befolyásolja a modellek komplexitása az eredményeket. Áttekintettem a modelleket, és rögzítettem mindegyik paraméterszámát abban bízva, hogy a paraméterszám szerinti intervallumokba való csoportosítás javít a bináris feladat megoldásának pontosságán. A modelleket paraméterszámuk alapján az alábbi csoportokba soroltam: kevesebb mint 5 milliárd, 6-10 milliárd, 10-130 milliárd, és több mint 130 milliárd paraméterrel rendelkező modellek. Az emberi osztállyal együtt ez egy 5 osztályos klasszifikációs feladatot eredményezett.

6 osztály használata

Ebben megközelítésben egy további intervallum került bevezetésre az alsó tartományban, finomítva a paraméterszám alapú csoportosítást, hogy pontosabban vizsgálhatóak legyenek a teljesítménykülönbségek. Ez lehetőséget biztosít annak ellenőrzésére, hogy bizonyos paraméterméretek – különösen a kisebb modellek — mennyire érzékenyek az árnyalt különbségek felismerésére. Az 5 milliárd alatti kategória két részre lett bontva, 1 milliárd paraméternél meghúzva a határt, ami remélhetőleg pontosabb elemzést tesz lehetővé a kisebb modellek skálájának hatásairól a bináris osztályozás teljesítményére. A hat különböző osztály az 5.3 ábrán tekinthető meg.

5.2.3. 3 osztály használata: generatív képesség szerinti csoportosítás

Bár a paraméterszám értékes mutató lehet a modell komplexitására, önmagában nem feltétlenül tükrözi teljes mértékben a modell teljesítőképességét. Feltételezhető. hogy további mutatók, például ranglistás helyezések és perplexitási pontszámok további betekintést nyújthatnak a szöveggenerálásra használt modellek hatékonyságába. Ezeknek a kiegészítő értékeléseknek a bevonásával célunk az osztályozás finomítása volt, hogy figyelembe vegyem azokat a különbségeket, amelyeket a paraméterszám esetleg figyelmen kívül hagy. Ez a megközelítés alapján egy ranglistaalapú ELO pontszámot is figyelembe veszek mint kiegészítő mutatót az értékelés teljesebbé tételéhez. Ehhez két ranglistát vettem figyelembe:

1. LLM explorer [1]
2. Chatbot arena [2]

[1] https://llm.extractum.io/list/?small
[2] https://lmarena.ai/

Míg az LLM Explorer ranglistán több modell szerepelt, a Chatbot Arena ranglistán csak néhány modell volt megtalálható, és mindössze négy modell jelent meg mindkét ranglistán. Ezekre a közös modellekre alapozva lineáris regressziót végeztünk a pontszámok összehangolása érdekében, így egy egységes skálát hoztunk létre az összes modell számára. Ez a lineáris regresszió az alábbi egyenletet eredményezte, ahol az LLM Explorer ranglistapontszámát jelöli:

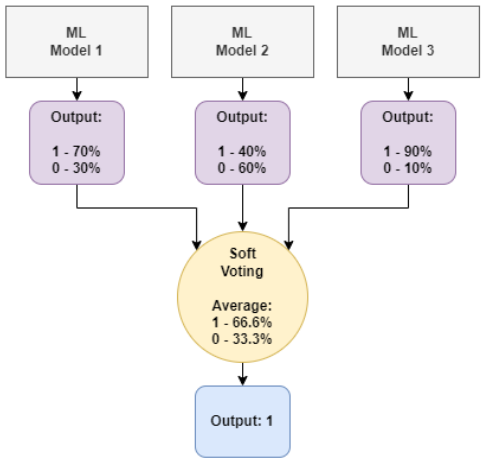
$$\text{ELO} = 1238x + 484$$

Ezzel a képlettel kiszámítottuk az ELO pontszámot minden modell számára. Egy küszöbérték került meghatározásra az osztályozáshoz: a 800-as ELO pontszám feletti modellek erős kategóriába lettek sorolva, míg a 800 pont alattiak a gyenge címkét kaptak. A 800-as határérték választása egy észrevehető különbség miatt lett kiválasztva, amely a 700 és 800 közötti mutatkozott, utalva arra, hogy ez a tartomány jelentős különbséget képviselhet a generáló képességekben. Azon 19 modell esetében, amelyek egyáltalán nem szerepeltek a ranglistákon, – egy kisebb utánajárást követően – az én saját megítélésem alapján raktam a megfelelő kategóriába. Ez a megközelítés végül háromosztályú feladatot eredményezett: *ember, gyenge, erős*. A végső csoportbontást az 5.3 ábra tartalmazza.

5.3. Ensemble módszerek

5.3.1. Standard Ensemble

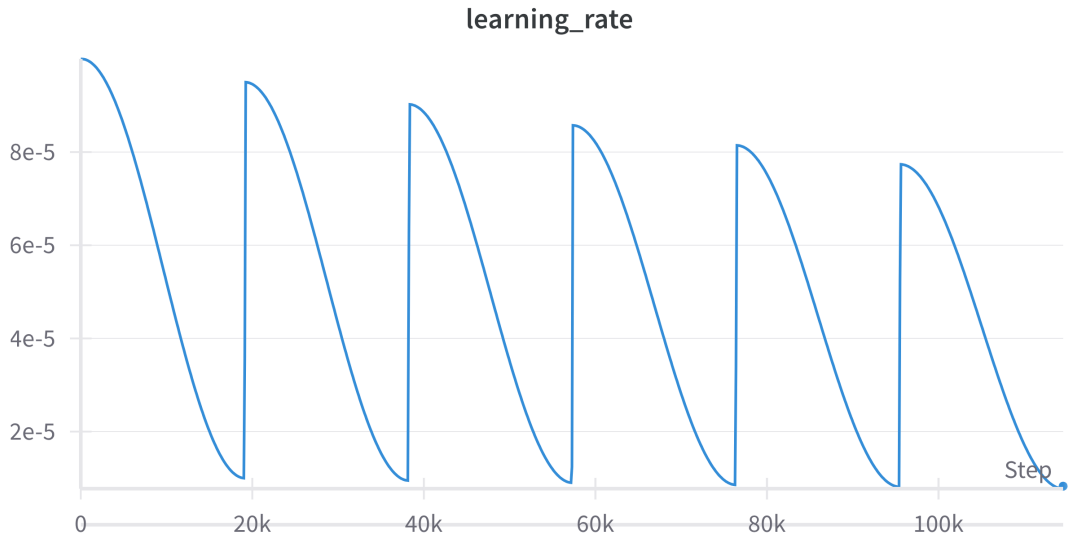
Minden lehetséges kísérletre meg lett valósítva ez a módszer, és kivétel nélkül az összesnél javított az eredményeken. Ez alapján általánosságban elmondható, hogy a standard ensemble javít az eredményen egy önálló modell teljesítményéhez képest. A legtöbb kísérletre három külön modellt tanítottam, és az egyes modelleket szavazással kombináltam, hogy a végső előrejelzés minél pontosabb legyen. Az 5.1 ábra szemlélteti 3 különálló modell szavaztatását.



5.1. ábra. Modellek szavaztatása Forrás: [15]

5.3.2. Snapshot Ensemble

A 3.3.2 fejezetben bemutatott snapshot ensemble eljárást konvolúciós hálókra bevezető cikk [7] alapján implementáltam a 41 osztályos konfigurációra azt az egyedi tanulási ráta ütemezést, amelynek viselkedése szinte teljesen megegyezik az eredetivel. Az 5.2 ábrán látható a ciklikus ismétlése az tanulási ráta emelésének és csökkenésének, amely segít időről-időre „kizökkenteni" a modellt. A kísérlet során összesen 6 modellt mentettem el, mindegyiket közvetlenül a tanulási ráta növelését megelőzően, így remélhetőleg mindig a lokális minimum környezetében sikerült a modellek kimentését megtenni. Ahhoz, hogy azonos, illetve összehasonlítható legyen a standard ensemble kísérlettel, 3 modellt kiválasztottam, így mindkét kísérlet 3-3 modellel hajtotta végre a szavazást. Ezt a 3 modellt úgy választottam ki, hogy megvizsgáltam az összes lehetséges modellhármast, lefuttattam mindegyiket a validációs halmazon, és ez alapján az együttműködésük alapján a legjobb eredményt elérő modelleket választottam ki. Az eredmények nagyon közel voltak egymáshoz, de megkaptam a legjobban együttműködő modellhármast: a 3., az 5. és a 6. tanulási ráta emelésnél mentett modellek. Ez egyébként némileg meglepő módon nem az a 3 modell volt, amelyek a legjobb eredményeket kapták külön-külön a validációs halmazra.



5.2. ábra. Tanulási ráta emelkedése és csökkenése

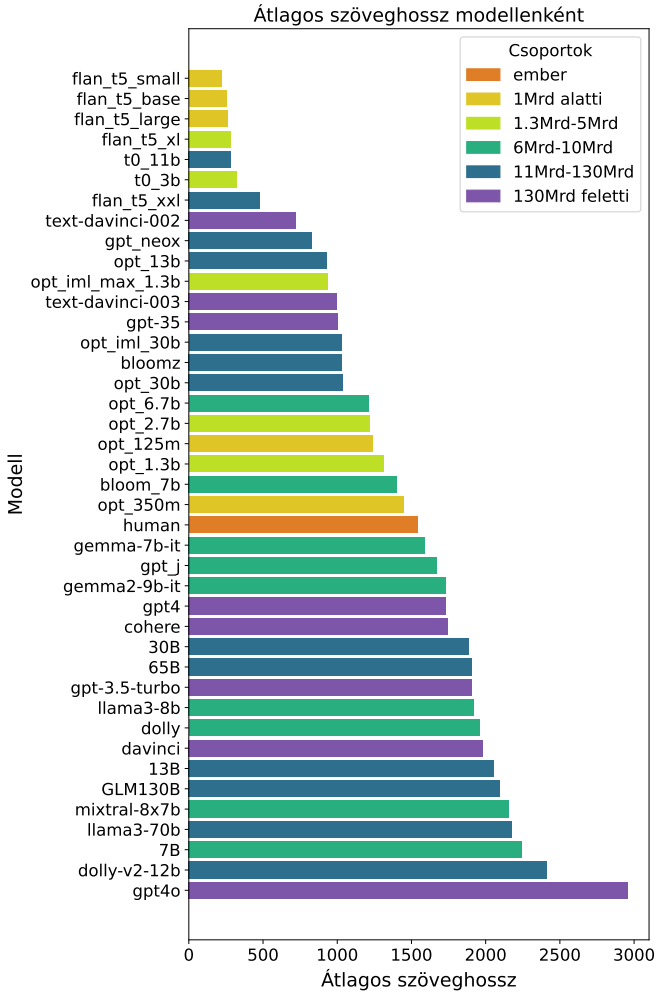
5.4. Hosszalapú szakértő modellek

A tanítóhalmazban található szövegek alapján megvizsgáltam az egyes modellek által generált szövegeket, hogy elemezzem őket, és különbségeket találjak közöttük. A legszembetűnőbb különbség a különböző modellektől származó szövegek hosszai között volt. Ahogy az várható, a nagyobb modellek, mint például a GPT-4, jelentősen hosszabb szövegeket állítottak elő. Ez az eltérés a szintetikus szövegek hosszában arra utal, hogy egy hosszalapú megközelítés alkalmazása javíthat az előrejelzés pontosságán. Ennek érde-

kében a modelleket három hosszalapú csoportba osztottam, mindegyiket egy speciális „szakértő” modell képviseli a rövid, közepes és hosszú szövegtartományokhoz:

- **Rövid szövegek**: [0, 745]
- **Közepes szövegek**: [746, 1613]
- **Hosszú szövegek**: [1614, 17445]

Ennél a kísérletnél az volt a feltételezés, hogy ez a struktúra pontosabb osztályozást eredményezhet, mivel kihasználja az egyes modellek hosszkategórián belüli jellemzőit, így egy robusztusabb és pontosabb módszert hozhat létre. Azt is érdemes megjegyezni, hogy a rövid szövegek felismerése nehezebb, mint a hosszabbaké [13]. Amennyiben egy rövid szövegről hozunk döntést, akkor a rövid szövegekre specializált szakértő modellnek vélhetőleg nagyobb esélye van megmondani, hogy milyen módon jött létre, mint egy olyan modellnek, amely mindenféle hosszúságú szöveget látott. A finomhangolást itt is 41-osztályos problémaként kezeltem. Az 5.3 ábrán látható az egyes modellek által generált szövegek hosszának az átlaga, ahol a ferde vonalazású minta jelöli az erős modelleket az egyes sávokban, illetve látható rajta a generatív modellek 6 osztályba történő beosztása is.



5.3. ábra. Szövegek hosszának átlaga modellekre nézve, ahol a vonalazott sávok az erős modelleket jelölik

5.5. Erős modellek kiaknázása

A verseny lezárulta után a teszthalmaz elemzése során észrevettem, hogy az kizárólag viszonylag új és erős nyelvi modellek által generált szövegeket, valamint ember által írt szövegeket tartalmazott. Ezzel szemben a gyengébb modellek által generált szövegek teljesen hiányoztak, ami jelentős eltérést mutatott a korábbi tanítási és validációs adathalmaz összetételéhez képest.

Ez a megfigyelés arra késztetett, hogy új megközelítést alkalmazzak a modellek tanításában és kiértékelésében. Úgy döntöttem, hogy kizárólag az erős nyelvi modellek által generált szövegekre és az ember által írt szövegekre fókuszálok a tanítási folyamat során. Ezáltal a tanítási adatokat jobban igazítottam a tesztadatok összetételéhez, abban bízva, hogy a modellek így pontosabban és hatékonyabban fognak teljesíteni a versenykörnyezetben.

Mivel ez már a verseny után történt, ennek a beadását nem lehetett kivitelezni, illet-

ve már nem is független a teszthalmaztól a kísérlet, mivel a változtatások éppen annak elemzéséből indultak ki. Ezek alapján nem feltétlen érdemes összevetni az eredményét a fejlesztési fázis alatt létrejött kísérletekkel, ámbár így sem haladta meg azok teljesítményét a verseny teszthalmazára. A megoldás 0,763 makró F1-pontszámot ért el. Ez már a szavaztatás eredménye, tehát a modellek külön-külön kiértékelve valószínűleg még ennél is rosszabbak.

6. fejezet

Eredmények

6.1. Saját eredmények

A verseny ranglistáján csak az utolsó feltöltés jelent meg, így a következőkben ismertetem, hogy mi alapján választottam ki a rangsorolásra szánt feltöltésemet. Az eredményeket makró F1-pontszámban mértem (hasonlóan a későbbiekben közölt eredményekhez). Ez a mérési metrika az az osztályonként számított F1-pontszámok egyszerű átlagát adja, így minden osztály egyenlő súlyt kap. Az F1-pontszám pedig a pontosság és a fedés harmónikus közepe, amelynek a képlete:

$$F1 = 2 \cdot \frac{\text{pontosság} \cdot \text{fedés}}{\text{pontosság} + \text{fedés}}.$$

Az első eredmény, aminél lehetett látni a modellek teljesítményét, az a validációs halmazon volt. Az elért értékeket a 6.1 táblázat mutatja.

Megoldás	**Szavaztatás nélkül**	**Szavaztatással**
Bináris	0,972	0,979
3 osztály	0,983	0,985
5 osztály	0,982	0,986
6 osztály	0,980	0,985
41 osztály	0,982	0,985
Snapshot	—	0,982
Hosszalapú	0,9675	—

6.1. táblázat. Eredmények a validációs halmazra (F1-pontszám)

Itt láthatóan az eredmények nem feltétlen adnak sok információt, mert nagyon jól teljesít az összes modell, és az értékek közel esnek egymáshoz. Ez alapján még nem nagyon lehetett meghatározni, hogy melyik megoldás is lenne a legjobb. Tekintettel arra, hogy az összes megoldás kb. 0,97-0,99 között van, egy korábban már általam vizsgált, ettől a versenytől független adathalmaz [17] segítségével igyekeztem döntést hozni. Ezzel az adattal már dolgoztam korábban, így már ismerős volt. Az erre kapott eredmények a 6.2 táblázatban találhatók.

Megoldás	Szavaztatás nélkül	Szavaztatással
Bináris	—	0,772
3 osztály	0,812	0,830
5 osztály	0,825	0,840
6 osztály	0,830	0,841
41 osztály	0,826	0,838
Snapshot	—	0,826
Hosszalapú	0,773	—

6.2. táblázat. Eredmények a független teszthalmazra (F1-pontszám)

Erre a teszthalmazra kapott eredmények már többet mondanak, jobban elkülönülnek az eredmények egymástól. Láthatóan a legjobb eredményt a 6 osztályos megoldás érte el, második az 5 osztályos, és a harmadik pedig a 41 osztályos megközelítés. Azon kívül általánosságban elmondható, hogy a legtöbb kísérlet erősen túlszárnyalta a bináris megoldást. A hosszalapú megoldás nem ért el túl jó eredményt, ez azért lehet, mert a szakértő modellek nem tudták felhasználni az erősségüket, mivel ebben a független teszthalmazban szinte kizárólag csak olyan szövegek voltak, amik a legrövidebb szövegeken tanított modellhez kerültek, ami azt jelenti, hogy csak egy modellhez kerültek a szövegek, mert a többi intervallum határát nem érte el. Ez alapján már kaptam egy képet, hogy valószínűleg a 6-osztályos megoldást érné meg utoljára feltölteni.

Még mielőtt döntöttem volna, egy újabb tesztet végeztem, mégpedig a fejlesztési fázis alatt a fejlesztők által kiadott teszt adathalmazon. Ez nagyjából 30,000 szöveget tartalmazott, hasonlított a tanítóhalmazra, így itt is viszonylag jó eredmények születtek. Ezt már csak kevesebb megoldással teszteltem, mert csak a beadott modell kiválasztásához kellett.

Megoldás	Szavaztatással
Bináris	0,844
3 osztály	0,915
5 osztály	0,905
6 osztály	0,897

6.3. táblázat. Eredmények a fejlesztési teszthalmazra (F1-pontszám)

A legjobb eredményt egyértelműen a 3 osztályos kísérlet érte el a 6.3 táblázatban, azonban a beküldési felület hibájából adódóan nem lehetett megváltoztatni az utolsó beadást a verseny végén. Ezáltal a véglegesen leadott megoldás a 6 osztályos maradt, és mint később kiderült, így jártunk jobban. A verseny vége után kiadták a teszthalmazt az osztálycímkékkel, így ki tudtam értékelni az összes megoldást. A baseline az 0,734 makró F1-pontszám volt, a módszere pedig egy RoBERTa [11] modell finomhangolása a bináris feladatra. A verseny teszthalmazán a különböző módszerek által elért eredményeimet a 6.4 táblázatban foglaltam össze.

Megoldás	Szavaztatás nélkül	Szavaztatással
Bináris	0,780	0,795
3 osztály	0,750	0,754
5 osztály	0,790	0,796
6 osztály	0,771	0,791
41 osztály	0,806	0,826
Snapshot	0,796	0,810
Hosszalapú	0,825	—

6.4. táblázat. Eredmények a verseny teszthalmazra (F1-pontszám)

Elég meglepő eredményt mutat a táblázat, tekintve, hogy a korábbi tesztek alapján ennek az ellentetjét láthattuk. Még a verseny szervezői által kiadott fejlesztési fázis alatt használható teszthalmazra kapott eredményre sem hasonlít. Megvizsgáltam a teszthalmazt, és az volt látható, hogy kizárólag erős, viszonylag új modellek által generált szövegek szerepeltek benne. Ez a tanítóhalmazban nem így volt. Ezt a változást a 3 osztályos megoldás sínylette meg a legjobban, ami bőven alulteljesítette a bináris megközelítést. A 6 osztályos szintén alulmaradt a binárishoz képest. Az 5 osztályos, illetve 41 osztályos snapshot ensemble módszer meghaladta, és valamennyivel jobb eredményeket adtak a kétosztályosnál. A legjobb eredményt a 41 osztályos kísérlet adta szavaztatással, ami magasan túlszárnyalta az összes megoldást. Emellett a hosszalapú egy elég nagy meglepetést okozott, mert a korábbi tesztek alapján nem adott jó eredményt, viszont a végső teszthalmazon rendkívül jól teljesített, és még szavaztatás nélkül is a második legjobb eredményt adta.

6.2. Online elérhető osztályozó eredménye

A saját kiemelkedőbb kísérleteimet összehasonlítottam egy másik, böngészőből is könnyen elérhető megoldással, a ZeroGPT-vel [1]. Pontos információt nem lehet találni arról, hogy milyen modellt és megoldást használnak, azonban a weboldalukon jó eredményeket ígérnek. Ezt a megoldást egy API híváson keresztül lehet elérni, és kéréseket küldeni. Ehhez felhasználtam a teszt adatbázisból véletlenszerűen 2000 adatot, és arra indítottam kiértékelést. Többre nem lehetett, mert az fizetős lett volna. Ezt persze megtettem a saját megoldásaimmal is, pontosan ugyanarra a 2000 szövegre. Az ő rendszerük úgy működik, hogy egy egész értéket ad vissza 0 és 100 között. Ez az érték azt jelöli, hogy mekkora eséllyel ember által írt a szöveg. Mivel így adta vissza az eredményt, húzni kellett egy vonalat, ami felett ember által készítettnek vettem a szöveget. Ez az érték 50 volt, amelyet zárt halmazként tekinthetünk, tehát ha valami 50 volt, az is beletartozott ebbe a kategóriába. Ezen küszöbérték módosítása nem igazán befolyásolja a modell eredményeit.

[1] `https://www.zerogpt.com/`

Megoldás	F1-pontszám
zeroGPT	0,715
3 osztály	0,767
5 osztály	0,814
6 osztály	—
41 osztály	0,836
Hosszalapú	0,845

6.5. táblázat. Teszthalmaz 2000 véletlenül kiválasztott szövegén elért eredmények (F1-pontszám)

A 6.5 táblázat alapján jól látható, hogy ez az online elérhető osztályozó nem ad feltétlenül jó eredményt, míg az én általam készített megoldások közül több is bőven meghaladja. A hosszalapú megközelítés itt nagyon jól teljesített.

A korábban ismertetett független teszthalmazt itt is használtam annak érdekében, hogy a versenytől egy teljesen független eredményt is láthassunk. Ehhez ebből az adathalmazból is véletlenszerűen kiválasztottam 2000 szöveget. Itt már csak az előzőleg legjobb (41 osztályos és hosszalapú) megoldásokat teszteltem a sajátjaim közül.

Megoldás	F1-pontszám
zeroGPT	0,615
41 osztály	0,831
Hosszalapú	0,807

6.6. táblázat. Független teszthalmaz 2000 véletlenül kiválasztott szövegén elért eredmények (F1-pontszám)

Érdekes eredmények születtek, mindegyik F1-pontszám valamennyivel rosszabb lett a 6.6 táblázat alapján, de a 41 osztályos megközelítés csak enyhén változott, míg a hosszalapú megoldás viszonylag sokat rontott. Emellett a zeroGPT eredménye jócskán rosszabb lett a korábbinál. Ezek alapján feltételezhető, hogy nem teljesen megbízható ez az online elérhető osztályozó, tekintve, hogy két független teszthalmazra is rosszabb eredményt adott. Lehetne az a feltételezés, hogy ez azért van, mert a teszthalmaz olyan adatokat tartalmaz, amely nem új modellektől van, mert amit a szöveg amit egy átlagos felhasználó fel akar ismerni, az valószínűleg valamely széleskörben ismert, jó eredményekkel rendelkező nagy nyelvi modelltől származhat. Ez az állítás valamennyire helytáll a független teszthalmazon, de az eredeti verseny teszthalmazán nem. Ott szinte kizárólag erősnek mondható, általában zárt forrású modellek voltak megtalálhatóak. Ezen információ tényében érdekes, hogy state-of-the-art modellek által generált szövegek esetében sem működik túl jól, mégis egy elég ismert, illetve sokat használt oldalról van szó, olyannyira, hogy 2024 októberében 18 millióan használták. [18]

7. fejezet

Összefoglalás, konklúzió

A kiindulási kérdésünk az volt, hogy megkülönböztethető-e gépi tanulási megoldásokkal az, hogy mesterséges intelligencia, vagy ember által készített szöveget látunk. A mai világban nagyon érzékeny terület a mesterséges intelligencia által generált tartalmak kezelése, és sok törvényhozás van alakulóban ezzel kapcsolatban [3]. Sok helyen előfordul a használata, gondoljunk a mindennapi életre, például az iskolákban is látható egy jelenség, hogy néhányan vizsgára, dolgozatra már ezekkel a generatív modellekkel készített szövegeket írják.

Az eredmények rámutatnak arra, hogy a bináris megközelítés nem feltétlenül a leghatékonyabb módja a gép által generált szövegek felismerésének. A feladat bináris problémaként való megközelítése helyett a feladat többosztályos problémaként történő modellezésre tettem javaslatot, mivel a különböző MI modellek által generált szövegek közötti különbségek megragadásával javítható a generált szövegek felismerésének minősége. Kísérleteim azt indikálják, hogy különböző megközelítések eltérően teljesítenek az egyes adathalmazokon: az egyik megoldás erős eredményeket mutatott a fejlesztési halmazon, míg egy másik a teszthalmazon teljesített kiemelkedően. Például a hosszalapú megközelítés a fejlesztési halmazon nem mutatott ígéretes eredményeket, azonban a teszthalmazon jól teljesített. A snapshot ensemble kissé jobb teljesítményt nyújt, mint egyetlen modell, miközben hasonló számítási hatékonysággal rendelkezik.

Érdekes módon a 3 osztályos megközelítés teljesítménye visszaesett a teszthalmazon, ami valószínűleg a gyengébb modellek hiányának tudható be ebben az adathalmazban. Ez arra utal, hogy bizonyos többosztályos konfigurációk kevésbé hatékonyak, ha csak magas teljesítményű modellekkel szembesülnek. Mindazonáltal, ezekkel a variációkkal együtt is, a 41 és az 5 osztályos megoldások felülmúlták a bináris megközelítést, és következetes előnyt biztosítottak a bináris modellezéssel szemben.

A saját modelljeim összehasonlítása a zeroGPT online osztályozóval szintén megerősítette, hogy az általam kidolgozott megoldások — különösen a 41 osztályos és a hosszalapú modellek — jól teljesítenek mind a verseny teszthalmazán, mind a független teszthalmazon.

A továbbiakban ezt a kutatást folytatni fogom, mivel további ötleteim is vannak, például ennek a problémának a tokenszintű vizsgálata, amely a bináris felismerés pontosságát tovább javíthatná.

Összegzésképpen elmondható, hogy a verseny során sikerült olyan modelleket kidol-

gozni, amelyek a mesterséges intelligencia által generált szövegek osztályozásának tekintetében sikeresek voltak. A megoldásaim azt mutatják, hogy a bináris osztályozáson túllépve a többosztályos stratégia -– különösen szavaztatás használatával — jobb eredményeket hozhat a gép által generált szövegek felismerésében.

Köszönetnyilvánítás

A dolgozat az Európai Unió támogatásával valósult meg, az RRF-2.3.1-21-2022-00004 azonosítójú, Mesterséges Intelligencia Nemzeti Laboratórium projekt keretében.

Irodalomjegyzék

[1] Megha Chakraborty és tsai. “Counter Turing Test (CT2): AI-Generated Text Detection is Not as Easy as You May Think - Introducing AI Detectability Index (ADI)”. *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Szerk. Houda Bouamor, Juan Pino és Kalika Bali. Singapore: Association for Computational Linguistics, 2023. dec., 2206–2239. old. DOI: 10.18653/v1/2023.emnlp-main.136. URL: https://aclanthology.org/2023.emnlp-main.136.

[2] Jacob Devlin és tsai. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. *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)*. Szerk. Jill Burstein, Christy Doran és Thamar Solorio. Minneapolis, Minnesota: Association for Computational Linguistics, 2019. jún., 4171–4186. old. DOI: 10.18653/v1/N19-1423. URL: https://aclanthology.org/N19-1423.

[3] European Parliament. *EU AI Act: First Regulation on Artificial Intelligence*. 2023. jún. URL: https://www.europarl.europa.eu/topics/en/article/20230601STO93804/eu-ai-act-first-regulation-on-artificial-intelligence.

[4] Gemini és tsai. *Gemini: A Family of Highly Capable Multimodal Models*. 2024. arXiv: 2312.11805 [cs.CL]. URL: https://arxiv.org/abs/2312.11805.

[5] Pengcheng He és tsai. “DeBERTa: Decoding-enhanced BERT with Disentangled Attention”. *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=XPZIaotutsD.

[6] Josh Howarth. *Number of Parameters in GPT-4 (Latest Data)*. 2024. aug. URL: https://explodingtopics.com/blog/gpt-parameters.

[7] Gao Huang és tsai. “Snapshot Ensembles: Train 1, Get M for Free”. *International Conference on Learning Representations*. 2017. URL: https://openreview.net/forum?id=BJYwwY9ll.

[8] Mihály Kiss és Gábor Berend. “Beyond Binary: Soft-Voting Multi-Class Classification for Binary Machine-Generated Text Detection Across Diverse Language Models”. *Proceedings of the 31st International Conference on Computational Linguistics (COLING)*. Abu Dhabi, UAE: Association for Computational Linguistics, 2025. jan.

[9] Weixin Liang és tsai. “Mapping the Increasing Use of LLMs in Scientific Papers”. *First Conference on Language Modeling*. 2024. URL: https://openreview.net/forum?id=YX7QnhxESU.

[10] Weixin Liang és tsai. “Monitoring AI-Modified Content at Scale: A Case Study on the Impact of ChatGPT on AI Conference Peer Reviews”. *Proceedings of the 41st International Conference on Machine Learning*. Szerk. Ruslan Salakhutdinov és tsai. 235. köt. Proceedings of Machine Learning Research. PMLR, 2024. 21–27 Jul, 29575–29620. old. URL: https://proceedings.mlr.press/v235/liang24b.html.

[11] Yinhan Liu és tsai. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. 2019. arXiv: 1907.11692 [cs.CL]. URL: https://arxiv.org/abs/1907.11692.

[12] Ilya Loshchilov és Frank Hutter. “Decoupled Weight Decay Regularization”. *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=Bkg6RiCqY7.

[13] OpenAI. *New AI Classifier for Indicating AI-Written Text*. 2023. jan. URL: https://openai.com/index/new-ai-classifier-for-indicating-ai-written-text/.

[14] OpenAI és tsai. *GPT-4 Technical Report*. 2024. arXiv: 2303.08774 [cs.CL]. URL: https://arxiv.org/abs/2303.08774.

[15] Ilyas Bin Salih. *What is Hard and Soft Voting in Machine Learning?* URL: https://ilyasbinsalih.medium.com/what-is-hard-and-soft-voting-in-machine-learning-2652676b6a32.

[16] Areg Mikael Sarvazyan, José Ángel González és Marc Franco-salvador. “Genaios at SemEval-2024 Task 8: Detecting Machine-Generated Text by Mixing Language Model Probabilistic Features”. *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*. Szerk. Atul Kr. Ojha és tsai. Mexico City, Mexico: Association for Computational Linguistics, 2024. jún., 101–107. old. DOI: 10.18653/v1/2024.semeval-1.17. URL: https://aclanthology.org/2024.semeval-1.17.

[17] Areg Mikael Sarvazyan és tsai. “Overview of AuTexTification at IberLEF 2023: Detection and Attribution of Machine-Generated Text in Multiple Domains”. *Proces. del Leng. Natural* 71 (2023), 275–288. old. URL: http://journal.sepln.org/sepln/ojs/ojs/index.php/pln/article/view/6559.

[18] SimilarWeb. *ZeroGPT Website Overview*. URL: https://www.similarweb.com/website/zerogpt.com/#overview.

[19] Statista. *Global AI Market Size - Forecast*. URL: https://www.statista.com/forecasts/1474143/global-ai-market-size.

[20] Ashish Vaswani és tsai. “Attention is all you need”. *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS’17. Long Beach, California, USA: Curran Associates Inc., 2017, 6000–6010. old. ISBN: 9781510860964. URL: https://arxiv.org/abs/1706.03762.

[21] Yuxia Wang és tsai. “GenAI Content Detection Task 1: English and Multilingual Machine-generated Text Detection: AI vs. Human”. *Proceedings of the 31st International Conference on Computational Linguistics (COLING)*. Abu Dhabi, UAE: Association for Computational Linguistics, 2025. jan.

[22] Yuxia Wang és tsai. “SemEval-2024 Task 8: Multidomain, Multimodel and Multilingual Machine-Generated Text Detection”. *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*. Szerk. Atul Kr. Ojha és tsai. Mexico City, Mexico: Association for Computational Linguistics, 2024. jún., 2057–2079. old. DOI: 10.18653/v1/2024.semeval-1.279. URL: https://aclanthology.org/2024.semeval-1.279.

[23] Zhuolin Yang és tsai. “On the Certified Robustness for Ensemble Models and Beyond”. *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=tUa4REjGjTf.