Fake News generation, and detection in articles, and the examination of related tools (2019 December)

Fake news hírek detekciója, generálása, és az érintett eszközök vizsgálata (December 2019)

Bálint Domonkos, Kolozsvári Dániel, Budapesti Műszaki és Gazdaságtudományi Egyetem

***Kivonat*— Az alábbi dokumentáció a 'Deep Learning a gyakorlatban Python és Lua alapokon' tárggyal kapcsolatos féléves munkánkat dokumentálja. A fake news hírek felismerésével, generálásával, a téma cutting edge eszközeivel foglalkoztunk. Az adataink a Massachusetts Institute of Technology egy kutatócsapata álltal összeállított és megcimkézett adathalmaz. Egy binárisan klasszifikáló ágenst készítettünk el, ami az igazi, és hamis hírek között tesz különbséget. Bidirectional LSTM neurális hálózatot használtunk a tanítás során. A dolgozat során részletesen taglaljuk a projekt indíttatását, az adatok megszerzését, tisztítását, analízisét, feldolgozását. Továbbá a rendszerterv részben taglaljuk az álltalunk használt háló álltalános működését, majd a tanítás konkrét megvalósítását, a tapasztalt problémákat, és megoldásokat, a hiperparaméter optimalizációt, a végső tanítást. Kitérünk továbbá az álltalunk vizsgált cutting edge, jelenlegi iparági sztenderd megoldásokra is, mint az ELMO, a BERT és a GPT2-es hálók.**

***Abstract*— This documentation is the summary of our work for the course "Deep learning in practise for the Python and Lua platoform". We were studying the generation, classification, aspects of fake news, as well as the cutting edge tools. We gathere our data from Massachusetts Institute of Technology research group's labeled database. Our basic agent is a binary classificator, which recognises fake, and reliable news. In our document we discuss how we gathered, cleaned, analyzed, prepared the data, the essence and architecture, of our neural networks, as well as the implementation of the training phase. We talk about our problems during the project, and the solutions we managed to come up with, the hyperparameter optimalization, and the final training phase. We talk about the cutting edge tools we managed to acquire, such as the BERT, ELMO, and GPT2.**

I. BEVEZETŐ

Kezdetben 3 fős csapatként megajánlott jegyet célzó házi feladatként azt a feladatot tűztük ki célul, hogy egy olyan ágenst hozunk létre, ami képes egy adott szó cikkről eldönteni, hogy fake news-e vagy sem. Tehát egy NLP (natural language processing) bináris klasszifikáló háló megtervezése, és impementálása volt a cél. Többféle megközelítéssel is próbálkoztunk, de végül az idő rövidsége, és egy csapattag elvesztése után Gyires-Tóth Bálint Tanár Úrral konzultálva a téma valamelyest módosult, azt a feladatot kaptuk, hogy járjuk körbe a fake news detektálás, és generálás témakörét, mind elméletben mind gyakorlatban. A mi saját munkánk, és a cutting edge megoldásokat is érdemes vizsgálnunk. A vizsgálat, és a próbák során Python 3-at, és a Keras frameworkot használtuk.

A dokumentációban először a tématerületet tágabban ismertetjük, ezután bemutatjuk a state-of-the-art megoldásoka, illetve az ezekkel szerzett tapasztalatainkat. A következő pontban leírjuk az adatbeszerzést, tisztítást, és az általunk létrehozott hálót tulajdonságait. Majd az Eredmények részben ismertetjük az egyes modelleken kapott eredményeket.

II. TÉMATERÜLET ISMERTETÉSE:

A 'fake news' fogalma sok területen sok jelentéssel bírhat, mi a következőt tartottunk a machine learning területén a leghasználhatóbbnak: 'a hamis hír (fake news) egy szándékosan, és bizonyíthatóan valótlan állításokat tartalmazó cikk, melyet valamilyen média felületen közölnek'. [1]

A szakirodalomban kétféle felosztást különböztetnek meg az egyes hamis hír detekciós módszerekre. Az egyik a fake news különböző aspektusai szerinti felosztás, a másik a konkrét használt technika szerinti felosztás. Az első alapján feloszthatjuk tudás alapú, stílus alapú, terjedés alapú, és megbízhatóság alapú módszerekre. A második felosztás kétféle technikát különböztet meg: a jellemző alapú (feature-based) és a kapcsolat alapú (relation-based). [1]

A tudás alapú megoldások közé tartoznak azok, amelyek valami tudástárral vetik össze a cikk állításait. [2] A stílusalapú módszerek ehelyett inkább az író szándékait vizsgálj, azt, hogy van-e félrevezető szándék a cikkben. [3] A terjedés alapú felismerés a hír online mozgásában keres mintákat [4], míg a megbízhatóság alapú módszerek azt feltételezik, hogy a honlapok linkhálója, illetve a kapcsolódó felhasználók ismeretségi, és tevékenységi

hálója alapján fel lehet ismerni ’fake-news’ klasztereket, és ez alapján azonosítani a hamis forrásokat. [5]
A deep learning megoldások mind jellemző alapúak. Mi a stílusalapú megközelítést használtuk, mivel úgy ítéltük a tudástárak folyamatos aktualizálása egyelőre nem megoldott, a cikkek terjedésére, és olvasóira vonatkozó adatok elérése pedig nem tűnt lehetségesnek.

III. KORÁBBI DEEP LEARNING MEGOLDÁSOK

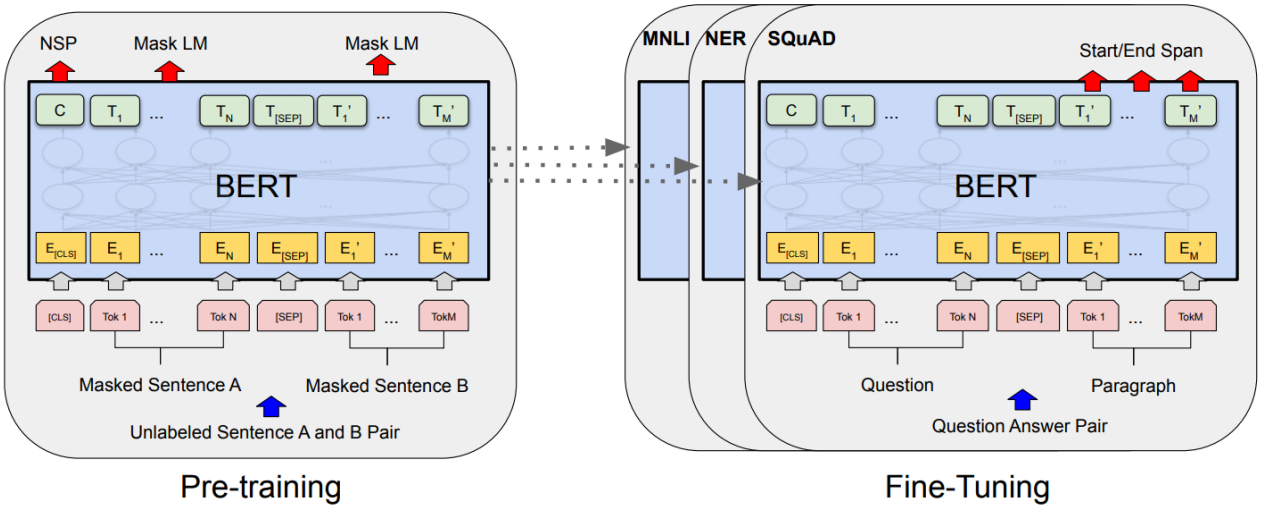
A korábbi megoldások és a munkánk szorosan összekapcsolódik, mivel a saját modell megalkotása mellett kipróbáltuk az elérhető generátor, és diszkriminátor modellek közül a jelenleg legjobbnak ítélteket. Ez négy különböző modellt jelent, melyek architektúráját, illetve a velük kapcsolatban tapasztaltakat foglaljuk össze a következő pontokban.

A) BERT

A BERT a Bidirectional Encoder Representation from Transformers rövidítése, melyet a Google fejlesztett [6].
A BERT háló tanítása két lépésből áll: az elő tanítás (’pre-training’), és a ’fine-tuning’. [ld. 1. ábra] Az elő tanítás során a modellt két különböző feladatra tanítják, nem címkézett adatokkal: az egyik a Masked LM feladat [7], másik a következő mondat predikció (’next sentence prediction). Az első során a bement bizonyos szavait maszkolják, és ezeket kell a hálónak kitalálnia a szókörnyezet alapján. A második feladat bináris osztályozás: egy mondat alapján kell a második mondatról a hálónak megállapítania, hogy az a következő mondat-e. A ’fine-tuning’ során a hálót a konkrét feladatra tanítják, mostmár címkézett adatokon, de azt az előtanítás során kapott súlyokkal inicializálják.
Nekünk az előtanított háló rendelkezésünkre állt, így csak ezt kellett tovább tanítani az előfeldolgozott adatokkal. Kétféle méretben érhető el a BERT háló, mi a kisebbiket használtuk, mivel ennek tanításához kevesebb erőforrás szükséges. A kisebb ’base’ hálónak 110 millió paramétere van, és 12 ’transformer block’ áll. A ’transformer’ egy enkóder-dekóder struktúrájú neurális háló, melyben fully connected rétegek, és attention rétegek váltogatják egymást [8]. Előnye, hogy képes szekvenciális adatokban nagy távolságra lévő kapcsolatokat is megtanulni illetve, hogy szekvenciális adatokat feldolgozása könnyebben párhuzamosítható a segítségével.
A háló pytorch megvalósítását használtuk, a külön erre készült könyvtár segítségével. Hogy a feladatunkra megfelelő legyen a BERT modellre egy dropout réteget, illetve egy egy rétegű sigmoid aktivációs fully connected réteget építettünk. Ezen rétegek szélességét az alattuk lévő rétegek szélessége meghatározta.
Tanítás során a bináris osztályozásnál standardnak mondható bináris keresztentriópia loss függvényt, illetve adam algoritmust használtunk.

B) GPT-2:

A GPT a Generative Pre-Trained Transformer rövidítése [9], melyet az OpenAI fejlesztett. Ahogy a nevéből is kitűnik ez lényeges hasonlóságokat mutat az előbb ismertetett BERT-el. Szintén két lépésben tanítják, egy felügyelet nélküli elő tanítás lépésben, és egy felügyelt fine tuning lépésben. Szintén transzformer blokkok képzik a háló alapját [ld. 2 ábra], de itt az eredetileg leírt egy módosított változatát használták. A BERT-hez képesti másik lényeges különbség, hogy itt balról jobbra haladva dolgozzák fel a bemenetet, összehasonlítva a BERT maszkolt nyelvi modell megközelítésével. Ennek megfelelően a BERT az egyes kimenetek kétirányú, a GPT pedig csak egyirányú (bal oldali) kontextus alapján tudja becsülni.
A GPT-2 az előbbi háló struktúráját követi, néhány változtatással [10]: a GPT-hez képest itt módosították a



1. ábra - A BERT tanításának két lépése

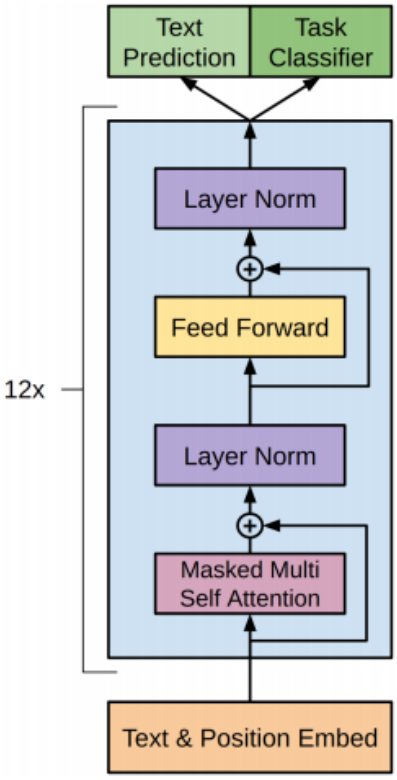
réteg normalizáció (’layer normalization’) helyét, az inicializáció módját, illetve a hálót méretét. Ez a háló már 1024 hosszú bemeneti szekvenciákat is képes kezelni, ami duplája a GPT bemenet méretének.
A GPT-2 modellt generálásra használtuk, egyrészt azért hogy megismerjük ezt a technológiát, másrészt hogy a kapott eredményekkel teszteljük a többi felismerő hatékonyságát. Összesen 4 féle méretű betanított háló adott ki az OpenAI (117M, 345M, 762M és 1542 millió paraméterrel rendelkező). Mi az összesnek a működését vizsgáltuk, több mintát csak a 762 millió súlyúval készítettünk, mivel ennek volt a legjobb a erőforrásigény/teljesítmény aránya. A hálót az oldalon közétettek alapján használtuk. A működéshez szükséges megadni egy szót, mondatot, amely alapján a háló generál szöveget. Mi összesen 10 BBC cikk összefoglaló alapján generáltunk 100 cikket.

C) RoBERTa:

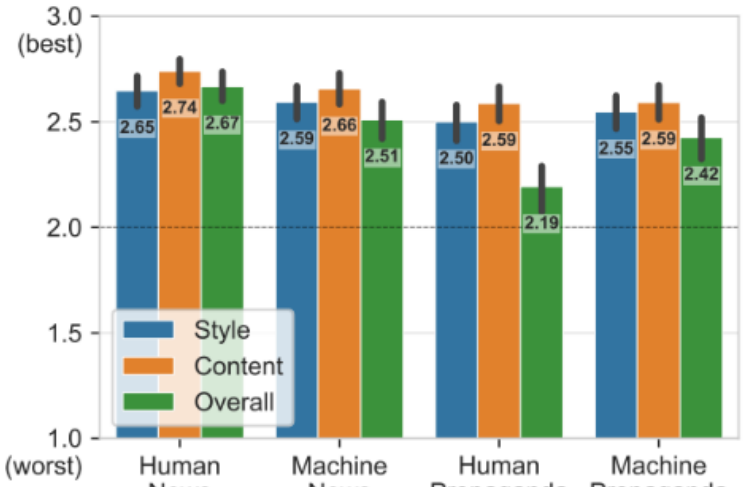
Sok GPT-2 generálta cikk igen magas hitelességi indexet (’credibility index’) ért el [11], ezért az OpenAI kiadott egy detektor modellt is. Ez nem a GPT-2-n, hanem egy másik modellen a Facebook által fejlesztett RoBERTa-n alapul. Ez a BERT-nél ismertetett architektúrájú háló, attól csak abban különbözik, hogy másként végezték az előtanítását. A BERT esetén azt állapították meg, hogy alul van tanítva, így a RoBERTa-t hosszabb ideig, nagyobb adathalmazon tanították, elhagyták a ’next sentence prediction’ (NSP) feladatot, és az MLM feladat során dinamikusan változtatták a bemeneti adatok maszkolását.
A detektor modellek közül a kisebbiket (12 rétegű ’base’ modellt) használtuk, megint csak az erőforrásigényességére való tekintettel. A felismerő egy alkalmazásba van csomagolva, mely UI-t is biztosít a használathoz, de a gyorsabb kiértékelésre érdekében ehelyett egy jupyter notebookos interfészt írtunk a felismerőnek.

D) Grover:

Kifejezetten a mesterséges (géppel alkotott) fake news felismerésére létrehozott modell [12]. Az volt az ötlet a modell megalkotásakor, hogy egy mesterséges álhír generátor tudná legjobban felismerni a mesterséges álhíreket.. Az eddigiekkel ellentétben ez a modell további metaadatokat is felhasznál: egy cikkek feletti domain, dátum, szerző, cím és cikk szöveg többváltozós valószínűségi eloszlást feltételez. A generáló háló feladata tehát az első négy metaadat alapján a cikk szöveget előállítani.
A modell a GPT-2 architektúráját követi, de egy a korábbinál nagyob úgynevezett RealNews adatbázison tanították. Ez a modell megveri a korábban említettek a saját maga által generált híreken. [12] Ez erősen torzítja persze az eredményt, de ugyanabban a cikkben készült egy összehasonlítás [3. ábra] a mesterséges, és valódi hírek emberek általi értékeléséről, és az látszik, hogy az emberek számára a Grover generálta propaganda hihetőbb, mint az ember által generált. Összességében tehát a Grover modell kimenetét minél pontosabb osztályozni releváns feladat.
A betanított generátor háló itt is elérhető volt több méretben (124M, 355M, 1500 millió paraméterű), mi a 124 millió súlyú ’base’ verziót használtuk. Mivel a generálásban van véletlen változó, ezért ugyanazon



2. ábra - A GPT transzformátor felépítése



3. ábra - Grover és az ember által generált szövegek összevetése

bemenetre más és más kimenetet ad az egyes esetekben. A generálást tehát 8 cikk alapján végeztük, összesen 80 cikket generáltunk belőlük. A diszkriminátor hálót nem publikálta az AllenAI, ezért ezt nekünk kellett tanítani. Végeredményben ez sajnos nem járt sikerrel: azt tapasztaltuk, hogy még a legkisebb háló esetén sem elégségesek az erőforrásaink, sem a rendelkezésünkre álló GeForce GTX 1070-es kártyának, sem a Collab által nyújtott Tesla K80-as kártyának nem volt elég a memóriája, hogy a tanítást elkezdjük.

IV. RENDSZERTERV

E) Kezdeti elgondolás

Kezdetektől fogva LSTM [13] (Long Short-Term memory) hálózatot terveztünk implementálni, az NLP (Natural Language Processing) megoldások kapcsán legtöbbször erre találtunk példákat, úgy éreztük a mi adatainkhoz kapcsolódóan ez a neurális háló típus a legcélszerűbb, a hosszú távú memóriája miatt. Mivel egy-egy szócikk igen hosszú lehet, az NLP elemzés alatt egy hagyományos rekurrens neurális hálózat esetén képbe kerülhet a vanishing gradient probléma, amit az LSTM háló kiküszöböl. A választásunk később a Bidirectional LSTM-re esett. Ez egy "kétirányú" LSTM háló, amely javíthatja a model teljesítményét klasszifikációs problémák esetén, szekvenciális idősorokban. A Bidirectional LSTM két LSTM hálót tanít, egyet a hagyományos módon, egyet pedig az adat szekvencia fordítottján. Ez a hálózat számára többlet értéket biztosíthat, és gyorsabb, teljesebb megértését segítheti az adott problémának.

V. MEGVALÓSÍTÁS

1. Adatok

A projekt ötlete úgy született, hogy láttuk, hogy a Massachusetts Institute of Technology-n (Massachusetts-i tudományegyetem) egy kutatócsoport nagyon széles spektrumú cimkézett hír adatbázist tett közzé. A hírek cimkéi között szerepelt megbízható, teljesen biztosan valótlan, cáfolható, tudományosan vitatható, stb cimkék, mi kiválasztottuk a biztosan valótlan, és a megbízható címkéket, és e szerint válogattuk ki a szó cikkeket az adatbázisból. Azért volt erre szükség, mert mi egy binárisan klasszifikáló ágenst szerettünk volna elkészíteni. A teljes adatbázis a filterezés után is 2.500.000 szó cikkből állt, ezt mi leredukáltuk 100.000 szócikkre kategóriánként.

A github maximum 100 megabyte-os adatméretet enged meg, így a repository-ban található bemeneti file-ainkat valamint az analitikát szolgáló file-okat is fel shardoltuk erre a méretre.

Az adatok előkészítése során először letöltöttük a teljes MIT adatbázist, majd a cikkek közül kiválogattuk azokat amelyek "fake news" vagy "reliable" cimkéjűek voltak, a többivel nem foglalkoztunk, eldobtuk őket. A következő lépésben a szétválogatott szócikkeket tartalmazó fileokat maximum 100 megabyte-os darabokra vágtuk, így már a githubon is kényelmesen tárolhatóak voltak. A következő lépés a fake , és a reliable cikkek közül kiválogatni a leggyakoribb 2000 szót, a későbbi analizálást elősegítendő, ezek után pedig eltávolítani mindkét listából azon szavak halmazát, ami mindkettő listában megtalálható. A végtermék ebben az esetben egy-egy szerializált lista volt a két cikk listában előforduló egyedi szavakból, valamint egy-egy lista a két cikkben előforduló (nem csak egyedi) szavakból, valamint a mellettük szereplő előfordulás gyakoriságai. Ezt követően a szöveg előkészítés fázisa következett. Az url-eket egy reguláris kifejezés segítségével az "url" szóra cseréltük a szövegben. A következő fázisban az egész szöveget kisbetűssé tettük, kitöröltük a mondatvégi írásjeleket, a vesszőket, és a dupla space-eket. A kezdeti tervezésnél úgy gondoltuk, hasznosak lehetnek ezek az információk, azzal az elő feltételezéssel élve, hogy a fake news cikkekre kevésbé jellemző a "minőségi újságírás" több indulatos, csupa nagybetűvel írt rész lehet rájuk jellemző, valamint kevésbé szabályos írásjel használat, de az adatokat megvizsgálva arra a következtetésre jutottunk, hogy nem ez a helyzet, így eltávolítottuk ezeket az adatokat. Ezek után a lemmatizáció, tehát a szótövezés következett. Ennek során arra törekedtünk, hogy az azonos szótövű szavakat azonos alakra is hozzuk, a model így könnyebben értelmezi őket azonosként. Ebben a spacy python modul volt segitségünkre. Ezután a stop wordök (azon szavak listája, amik ebben az értelemben "töltelék" szerepet látnak el, nem kontribútálnak konkrétan a szöveg értelmezéséhez egy neurális háló szempontjából, például "is", "a", "the" stb) kifilterezése következett, ezeket is eltávolítottuk a szövegekből, egyszerű space-ekkel helyettesítve őket. Ezzel az adatok megszerzése, vizsgálata, előkészítése be is fejeződött.

a. 2. Konkrét megvalósítás

A tanítást megelőzően, az adataink memóriába olvasásával kezdünk. Ezt követően meghatározzuk a tanítási intervallumot, majd pedig a kiválasztott szócikk listák elemeit ellátjuk címkékkel, amik jelzik, hogy igazi megbízható, vagy pedig hamis hírek. Ezt követően elkészítettük a tréning, validációs, és teszt adatsorokat, 0.2, és 0.1 súlyokkal. Következett a beágyazási mátrix elkészítése, ami egy facebook álltal előtanított FastText [14] szó beágyazási mátrixon alapult. Azért ezt alkalmaztuk mert ez képes a beágyazás szótárában nem szereplő szavakhoz is valamilyen szemantikailag

értelmes vektort rendelni, azáltal hogy vizsgálja a szóban előforduló ngrammokat. Ezt követően az adatokhoz tartozó labeleket vektorrá alakítottuk, hogy a tanítás során is használhatóak legyenek. Ez után következett a bidirectional LSTM definiálása.

A neurális hálózatunk a következő rétegekből áll: egy bemeneti réteg, a bejövő adat formájától függően, két beágyazási réteg, ahol megkapja a réteg az előzőleg elkészített beágyazási mátrixot, valamint egy dropout réteg, egy bidirectional LSTM réteg, és a három output réteg, egy rectified linear unit (relu) aktivációs függvénnyel rendelkező Dense réteg, egy dropout réteg, és egy softmax aktivációs függvénnyel rendelkező Dense réteg. Veszteségfüggvénynek bináris kereszt entrópiát használtunk, ez a bináris klasszifikáció esetén igen sűrűn használt veszteség függvény típus. A vizsgált metrikák pedig a model predikciós pontossága. A modell teljes paraméter számossága több mint 3 millió paraméter volt, amik közül mind trainable parameter volt, azaz tanítható paraméter.

b. 3. Problémák

Már az első tanítás során kiderült, hogy problémánk van az adatokkal kapcsolatban. A teljes adathalmazon tanítva a hálózat 10 epoch után 93.81%-os pontosságot produkált, ami alapján azt a következtetést vontuk le, hogy az adatok között maradhattak olyan információk, amikre a hálózat könnyedén rátanul, és ezek alapján következtetve magas százalékban tud prediktálni a mi adathalmazunk esetén. Ez azért jelenthet problémát, mert ha esetleg azokat a mintázatokat, amelyekre valóban szükséges lenne rátanulnia, nem, vagy kisebb mértékben veszi figyelembe, és ha kilépünk a mi adataink köréből, és egy más adathalmazból választunk cikkeket, rosszabbul fog teljesíteni a hálózat. Ezért elkezdtük keresni a probléma okát, a már korábban létrehozott adat analitikát vizsgáltuk meg. Úgy gondoltuk, hogyha van olyan információ amire a hálózat ilyen könnyen rátanul, akkor ez kikövetkeztethető a gyakori szavak listájából, és így is lett. A fake news listában magas helyen szerepelt a bitcoin, blockchain szavak, és észrevettük hogy bizonyos cikkekben vannak headline-ok ehhez köthetően. Ezektől megtisztítottuk az adathalmazunkat, és újra tanítottuk a hálót. Sajnos, bár ezt a plusz információ átadást sikerült kiküszöbölnünk a hálónkból, még mindig túl jól klasszifikál a háló. Két lehetőség maradt: valóban ennyire jó eredményt ér el a háló 10 epoch alatt, vagy maradt még az adatokban nem oda illő információ. Az adatok további analízise során sajnos nem találtunk további anomáliákat, így a meglévő adatunkon próbáltuk meg a paraméter optimalizáció lépéseit elvégezni.

c. 4. Hiperaraméter Optimalizáció

Hiperparaméter optimalizáción azt a folyamatot értjük, amikor azokat a paramétereket, amelyeket a hálózat nem tud megtanulni, tehát egy külső inputkét érkezik, megpróbáljuk úgy meghatározni, hogy a hálózat betanulása szempontjából a legkedvezőbb értéket vegyék fel. Tehát a hiperparaméter térben keressük azt a kombinációt, ami számunkra a legkedvezőbb. Mi 2x2-es (viszonylag kicsinek számító) Hiperparaméter térrel végeztük az optimalizációt. A hiperparaméter tér az adam optimalizátorból, az rmsprop optimalizátorból, a 0 dropout értékből, és a 0.5 dropout értékből, pontosabban ezek kombinációjából áll. A hiperparaméter optimalizáció lényegében azt jelenti, hogy minden kombináció esetén futtatunk egy tanulást kisebb mintán, majd a legpontosabb, legjobbnak értékelt kombinációjú paraméterekkel lefuttatjuk a végső tanítást a teljes mintán, és így kapjuk meg a végső binárisan klasszifikáló neurális hálónkat.

Miután az összes lehetőségen végigmentünk a végső eredmény szerint a legjobb paraméter kombináció az RMSPROP optimalizátor, és a 0.5-ös dropout érték volt. Ezekkel a paraméterekkel tanítottuk a végső elkészült hálónkat.

d. Új szóbeágyazás:

Az eddigek során alkalmazott FastText szóbeágyazásnak az a problémája, hogy a beágyazás tanítása után már nem számít a szó kontextusa, így az azonos alakú, de különböző jelentésű szavakhoz is azonos vektort rendel.

Ehelyett kipróbáltunk egy konextus függő szóbeágyazást a ELMot (Embeddings from Language Models) [15]. Itt a szóbeágyazás az egész mondat függvénye: az egyes szavak vektora két kétirányú LSTM réteg belső állapotának lineáris kombinációjaként jön létre.

Az ELMo angolra előretanított változat elérhető TensorflowHubon, így nekünk csak ezt kellett a korábbi beágyazási réteg helyére betenni.

Az ELMo számítás nagyon erőforrás igényesnek bizonyult, és a háló tanítása során rendre resource overflow üzeneteket kaptunk, így le kellett csökkentenünk a maximális mondat hosszat 512-re, ezáltal a bemenet a korábbiakhoz képest megfeleződött.

VI. EREDMÉNYEK

Kétféle feladatra értékeltük ki az egyes hálókat: egyrészt, hogy mennyire jól tudják az eredeti adathalmazból kiválogatott hamis, és igaz híreket megkülönböztetni (fake vs real), másrészt pedig, hogy a generált hamis híreket hogyan tudják megkülönböztetni a valósaktól(machine vs real). Ehhez két teszthalmazt definiáltunk, az egyik 10 000 cikk volt az eredeti adathalmazból, ebből közel fele-fele igaz és hamis. A másik teszthalmaz a 80 darab Grover generálta, 100 darab GPT-2 generálta és 180 darab igazi hírből áll.

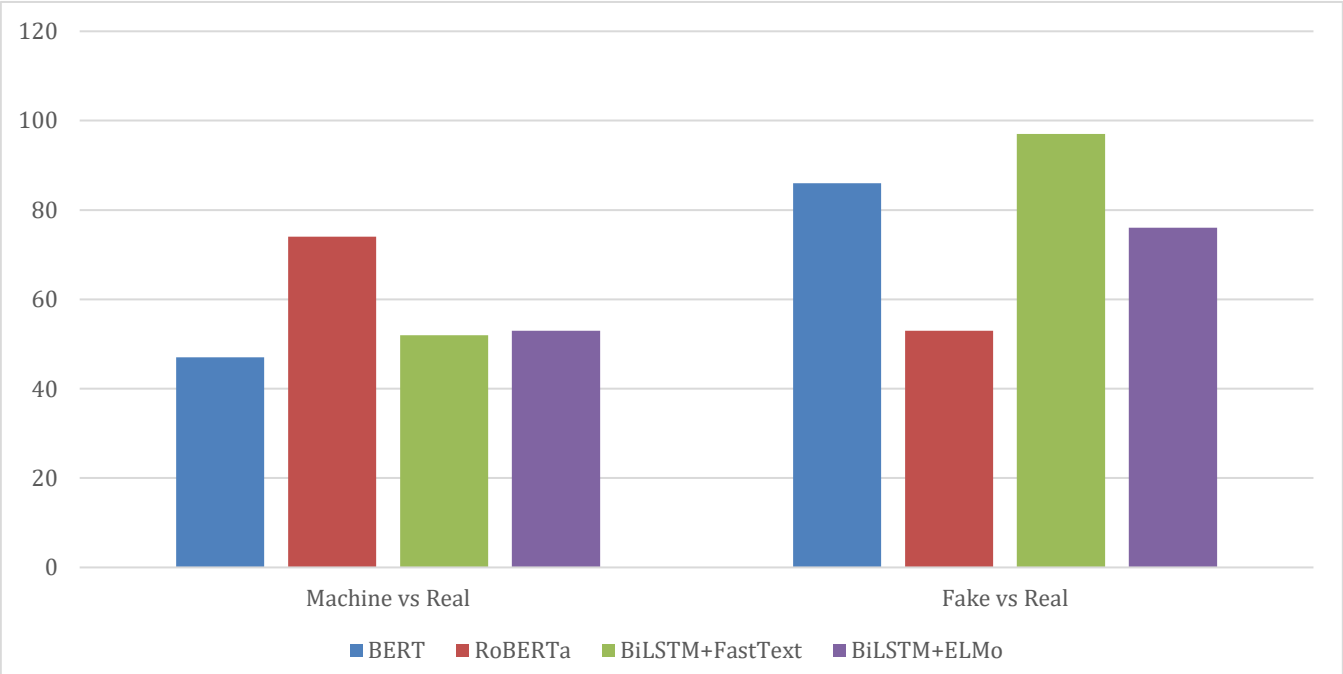
A két halmazzal négy különböző modellt teszteltünk: a FastText+BiLSTM az ELMo+BiLSTM a RoBERTa és a BERT modelleket. A kapott eredményeket mutatja a [4. ábra]. Az a tapasztalat, hogy a felvázolt két feladat teljesen különbözi, az egyik feladatra tanított háló nem képes a másikat jó pontossággal felismerni. A RoBERTa volt egyedül a machine vs real feladatra tanítva, viszont ez sem ismerte fel jól a Grover hálóval előállított mesterséges álhíreket. Ha ezeket kihagytuk a BERT kimenetét, és az igazi híreket 95%-os pontossággal ismerte fel.

VII. JÖVŐBELI TERVEK

Továbbfejlesztési ötletként elsődlegesen az eredeti ötletünk merült fel, mégpedig az ágens deployolása live service-ként. Ez úgy valósulhatna meg, hogy beágyazzuk a neurális hálót egy webalkalmazásba, ahol meg lehet adni neki egy cikk URL-jét. Az alkalmazás letölti az oldal tartalmát (esetleg adattisztítást is végez) majd pedig prediktál a neurális háló alapján. A felhasználó ezek után elbírálja hogy jól tippelt-e a neurális háló, és visszajelzést is adhat, ami után a háló tud tovább tanulni.
Természetesen ami talán még fontosabb továbbfejlesztési irányvonal, az az adatok megtisztítása, hogy a háló teljesítménye értékelhető legyen “valós életbeli” cikkek esetén is, ehhez pedig az szükséges, hogy megtaláljuk az adatokban rejtőző plusz információt, amire rá tud tanulni az álltalunk készített ágens.

VIII. ÖSSZEFOGLALÁS

A házi feladat elvégzése során az előadásokon, és gyakorlatokon megismert machine learning, és deep learning módszereket használtuk, majd ezeken tovább lépve, a cutting edge megoldásokon tudtuk bővíteni a látókörünket. Kezdésként az adatbeszerzéssel, és tisztítással foglalkoztuk, majd az adatok előkészítése után elkezdtük tanítani a hálónkat. Észrevettük, hogy az adatok között valami olyan információt adunk át a hálózatnak, amivel nagyon kevés epoch alatt nagyon jó teljesítményt ér el, ezért további analítis után kivágtuk az egyes cikkekben szereplő headlineokat. Ez után sem csökkent a hatékonyság, ez után tehát időszűkében ezekkel az adatokkal végeztük el a hiperparaméter optimalizációt. Az elkészült modell-t teszteltük különféle generált cikkekkel.



4. ábra - Az egyes modellek pontossága a gépi cikkek és valós cikkek (bal oldal), illetve az igaz és hamis cikkek feladaton

IX. HIVATKOZÁSOK

[1] X. Z. a. R. Zafarani, „Fake News: A Survey of Research, Detection Methods, and Opportunities,” *arXiv,* 2018.

[2] X. L. D. a. E. G. a. G. H. a. W. H. a. N. L. a. K. M. a. T. S. a. S. S. a. W. Zhang, „Knowledge Vault: A Web-Scale Approach to Probabilistic Knowledge Fusion,” in *The 20th {ACM} {SIGKDD} International Conference on Knowledge Discovery and Data Mining*, New York, 2014, pp. 601--610.

[3] G. D. B. a. R. D. H. a. J.-A. L. E. a. L. F. S. a. O. N. G. a. S. C. M. a. K. W. M. a. E. C. C. a. R. Rustige, „Lyin' Ted’, ‘Crooked Hillary’, and ‘Deceptive Donald’: Language of Lies in the 2016 US Presidential Debates,” 2017.

[4] J. Ma, „Rumor Detection on Twitter with Tree-structured Recursive Neural Networks,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia, 2018, pp. 1980-1989.

[5] D. a. Z. L. a. K. J. a. K. I. D. Zhang, „What Online Reviewer Behaviors Really Matter? Effects of Verbal and Nonverbal Behaviors on Detection of Fake Online Reviews,” *Journal of Management Information Systems,* pp. 456-481, 2016.

[6] J. D. a. M.-W. C. a. K. L. a. K. Toutanova, „BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” *arXiv,* 2018.

[7] W. L. Taylor, „“Cloze Procedure”: A New Tool for Measuring Readability,” *Journalism Quarterly,* 1953 .

[8] A. V. a. N. S. a. N. P. a. J. U. a. L. J. a. A. N. G. a. L. K. a. I. Polosukhin, „Attention Is All You Need,” *arXiv,* 2017.

[9] A. Radford, „Improving Language Understanding by Generative Pre-Training,” 2018.

[10] A. R. a. J. W. a. R. C. a. D. L. a. D. A. a. I. Sutskever, „Language Models are Unsupervised Multitask Learners,” 2019.

[11] I. S. a. M. B. a. J. C. a. A. A. a. A. H.-V. a. J. W. a. A. R. a. G. K. a. J. W. K. a. S. K. a. M. M. a. A. N. a. J. B. a. K. M. a. Jasmi, „Release Strategies and the Social Impacts of Language Models,” *arXiv,* 2019.

[12] R. a. H. A. a. R. H. a. B. Y. a. F. A. a. R. F. a. C. Y. Zellers, „Defending Against Neural Fake News,” in *Advances in Neural Information Processing Systems 32*, 2019.

[13] S. a. S. Hochreiter, „Long Short-Term Memory,” *Neural Comput,* 1997.

[14] T. a. G. E. a. B. P. a. P. C. a. J. A. Mikolov, „Advances in Pre-Training Distributed Word Representations,” in *Proceedings of the International Conference on Language Resources and Evaluation*, 2018.

[15] M. E. a. N. M. a. I. M. a. G. M. a. C. C. a. L. K. a. Z. L. Peters, „Deep contextualized word representations,” in *Proc. of NAACL*, 2018.