Névelemfelismerés előtanított BERT neurális hálózat segítségével

Szöveg- és webbányászat
szorgalmi feladat

Dokumentáció

Szerző: Füleki Fábián – AEP0TG

Motiváció

A Named Entity Recognition (NER) a Natural Language Processing (NLP) egyik alapvető feladata, melyre sokféle megoldási módot fel lehet lelni. Személyes érdeklődésem miatt feltétlenül egy deep learning alapú rendszert szeretnék megalkotni, amely nem mellesleg egybevág az elmúlt évek technológiai fejlődésével is.
Két évvel ezelőtt részt vettem egy NLP workshopon, amelyet Dr. Gyires-Tóth Bálint tartott az Nvidia képviseletében, azonban az ott elsajátított tudás csak az alapokat fedte le. Azóta aktívan nem foglalkoztam semmivel szövegfeldolgozási témában, viszont figyelemmel követtem a különböző fejlesztéseket, mint például a Google AI által fejlesztett Bidirectional Encoder Representations from Transformers (BERT) név alatt futó megközelítést (lásd https://arxiv.org/abs/1810.04805). A BERT self-supervised módon történő tanítása szintén érdekes számomra, mivel jelenleg self-supervised depth estimation-el foglalkozom az önálló labor keretein belül. További érdekesség, hogy a transformer neurális hálókat elkezdték képfeldolgozásra is használni, lásd https://arxiv.org/abs/2010.11929 .

A feladat részletei

A Named Entity Recognition a korpuszban előforduló tulajdonnevek, objektumok, időpontok, stb felismeréséből álló feladatkör. Ebben a szorgalmi feladatban a következő szóosztályok felismerése a cél:
- esemény
- földrajzi entitás
- geopolitikai entitás
- objektum, műtárgy
- szervezet
- személy
- idő

Az össze többi szó az "egyéb" osztályba tartozik. A jelenlegi feladat kihívása abba rejlik, hogy az egyes névelemek nem csak egy szóból állhatnak, és ezt dedikáltan jelezni is kell a kimeneten. Ez még jobban diverzifikálja a lehetséges kimeneteket, így végül 15 féle osztályba lehet majd sorolni az egyes szavakat.

A BERT architektúra és tanítása

A BERT egy transzformer neurális hálózat alapú megoldás, amely képes a szavakat a teljes mondattal együtt értelmezni, szemben a korábban széles körben elterjedt megoldásokkal (például LSTM) szemben. A BERT esetében a bemenet egy tokenekből álló sorozat, amely beágyazáson (WordPiece) esik át.
A BERT szerzői a neurális hálót két fázisban tanították. Az első fázis a Masked Language Model, amiben a bemeneten található egyes szavak 15%-os valószínűséggel ki vannak

maszkolva. Ebben a fázisban a cél ezeknek a kimaszkolt szavaknak a helyreállítása a mondat megmaradt részeinek a segítségével.
A második fázis a Next Sentence Prediction, ahol a model megpróbálja megállapítani, hogy a bemenetként kapott két mondat egymást követik-e az eredeti korpuszban.

A feladat megoldásának részletei

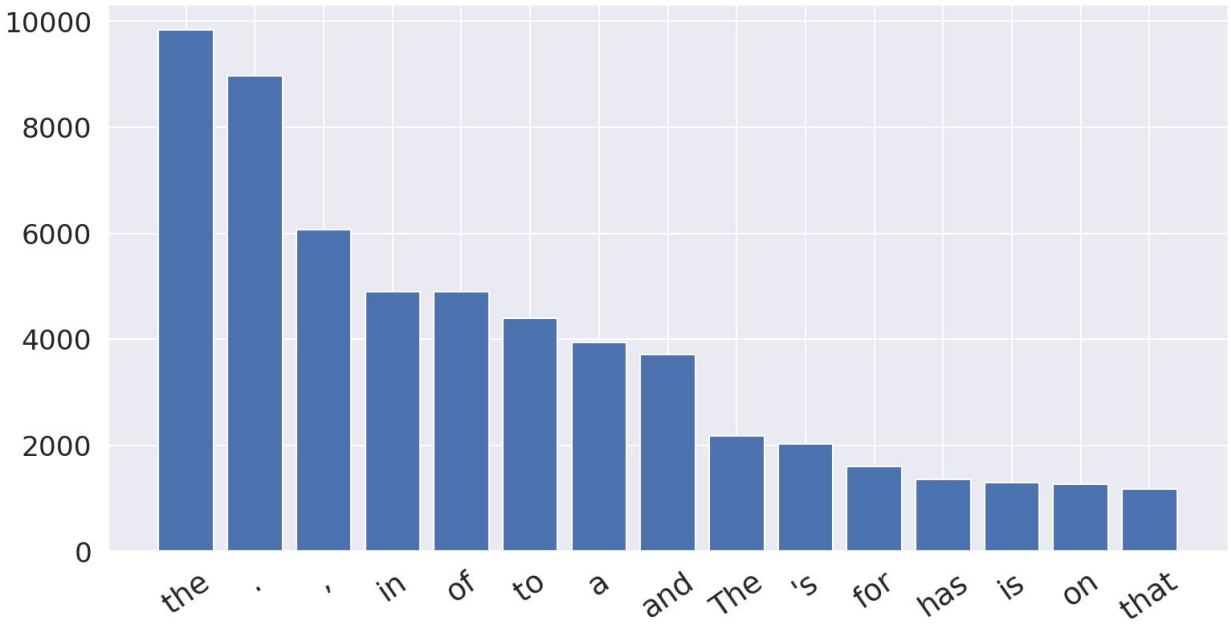
Mint minden eddigi deep learning alapú megoldást, ezt is a Python programozási nyelv segítségével készítettem el. Az adathalmaz megismeréséhez és az előfeldolgozáshoz a pandas, illetve a transformers csomagokat használtam. A tanítást a Pytorch csomag segítségével végeztem.
Ebben a dokumentációban nem részletezem az implementáció részleteit, csak a fontosabb lépéseket és kimeneteket mutatom be és magyarázom meg.

Az adatok megismerése

Minden adatfeldolgozási feladatnál fontos a kapott adatok formátumának, tartalmának, minőségének megismerése, így én is elvégeztem ennek különböző lépéseit.
Első körben megvizsgáltam a kiadott csv állományokat, ahol azt tapasztaltam, hogy a tesztfájloknál hiányoznak a fejlécek. Az egyszerűség kedvéért ezeket a fájlokban pótoltam.
Ezután beolvastam az egyes fájlokat, és megvizsgáltam, hogy van-e bennük hiányzó adat. Szerencsére az összes fájl összes sora tartalmazza a megfelelő adatokat:

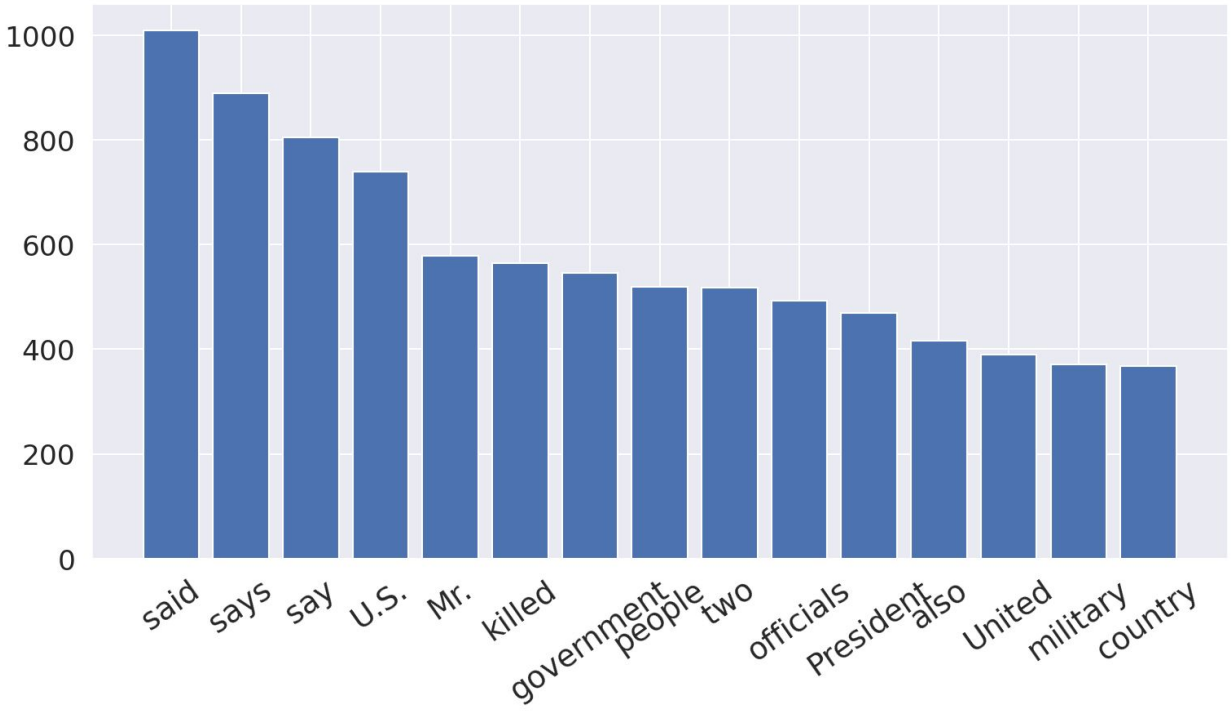
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 196646 entries, 0 to 196645
Data columns (total 4 columns):
 #   Column      Non-Null Count   Dtype
---  ------      --------------   -----
 0   Sentence #  196646 non-null  object
 1   Word        196646 non-null  object
 2   POS         196646 non-null  object
 3   Tag         196646 non-null  object
dtypes: object(4)
memory usage: 6.0+ MB
```

Következő lépésként megvizsgáltam a korpusz szavainak eloszlását:



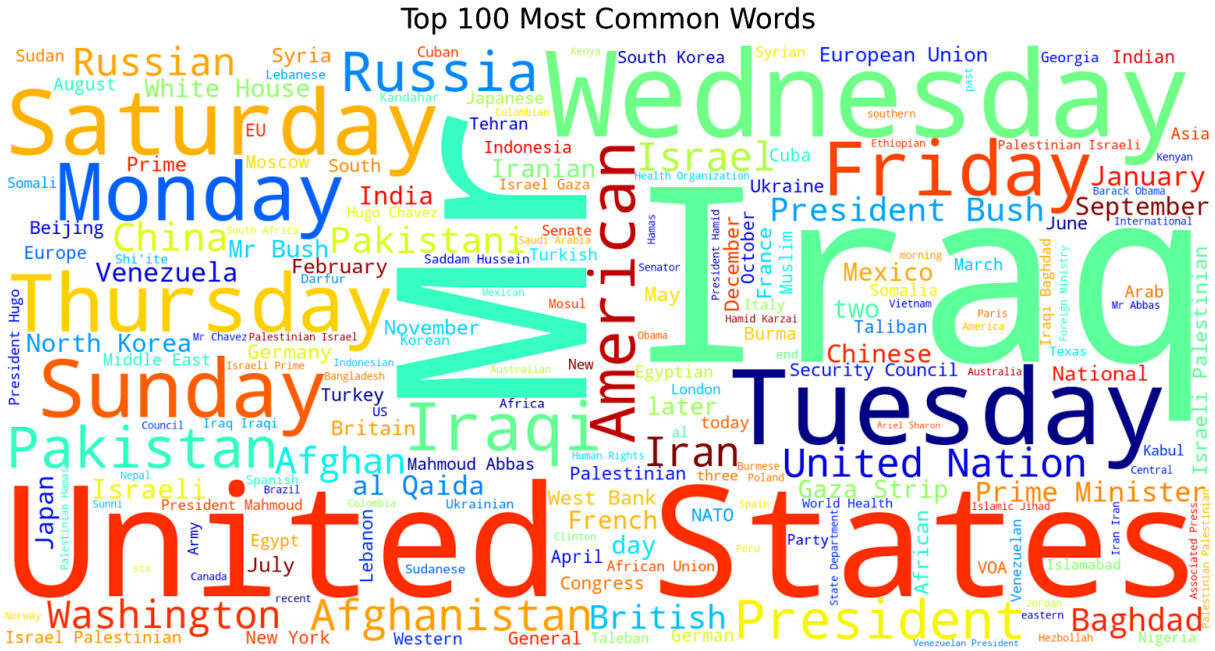
A szöveg nyelve tehát angol, és nem meglepő módon a “the” határozott névelő fordul elő benne leggyakrabban. Kiderül továbbá, hogy a mondatokban található központozási jelek külön szóként vannak kezelve.

A stopszavak eltávolításához az nltk csomagot használtam és miután eltávolítottam őket a korpuszból, a következő hisztogramot kaptam:



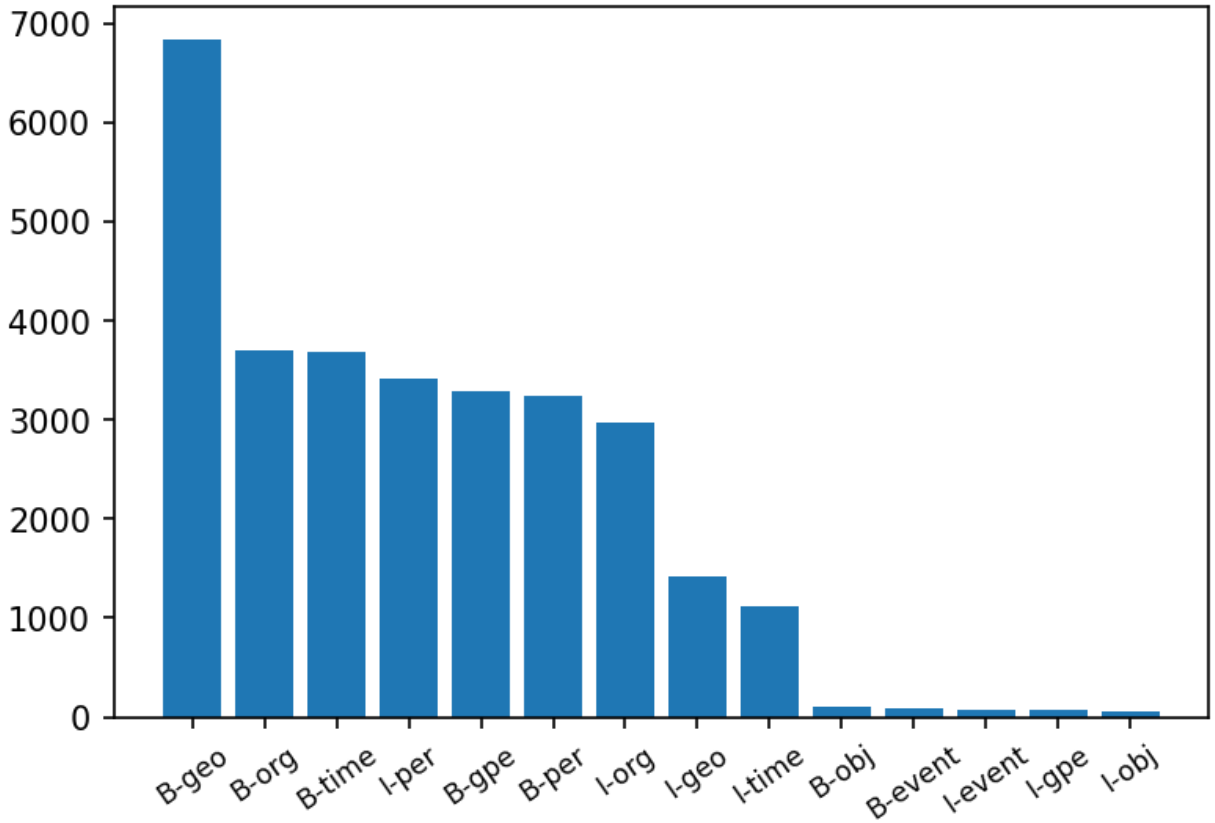
Itt már kezd látszani, hogy a korpusz erősen politikai irányultságú mondatokat tartalmaz.

Ezután kiszűrtem az “other” kategóriába sorolt szavakat és szófelhő formában vizualizáltam őket



Ez az ábra mutatja meg, hogy a szövegbázis teljes egészében geopolitikai témájú.

Megvizsgáltam az adatbázisban található osztályok eloszlását is:



Jól látható, hogy az osztályok erősen kiegyensúlyozatlanok. Az ”other” osztály önmagában kiteszi az esetek 85%-át.

Ezután megvizsgáltam, hogy az egyes osztályokban mik a leggyakrabban előforduló kifejezések. Ehhez bent hagytam a stopszavakat is, mivel azok lehetnek egy kifejezés részei is.

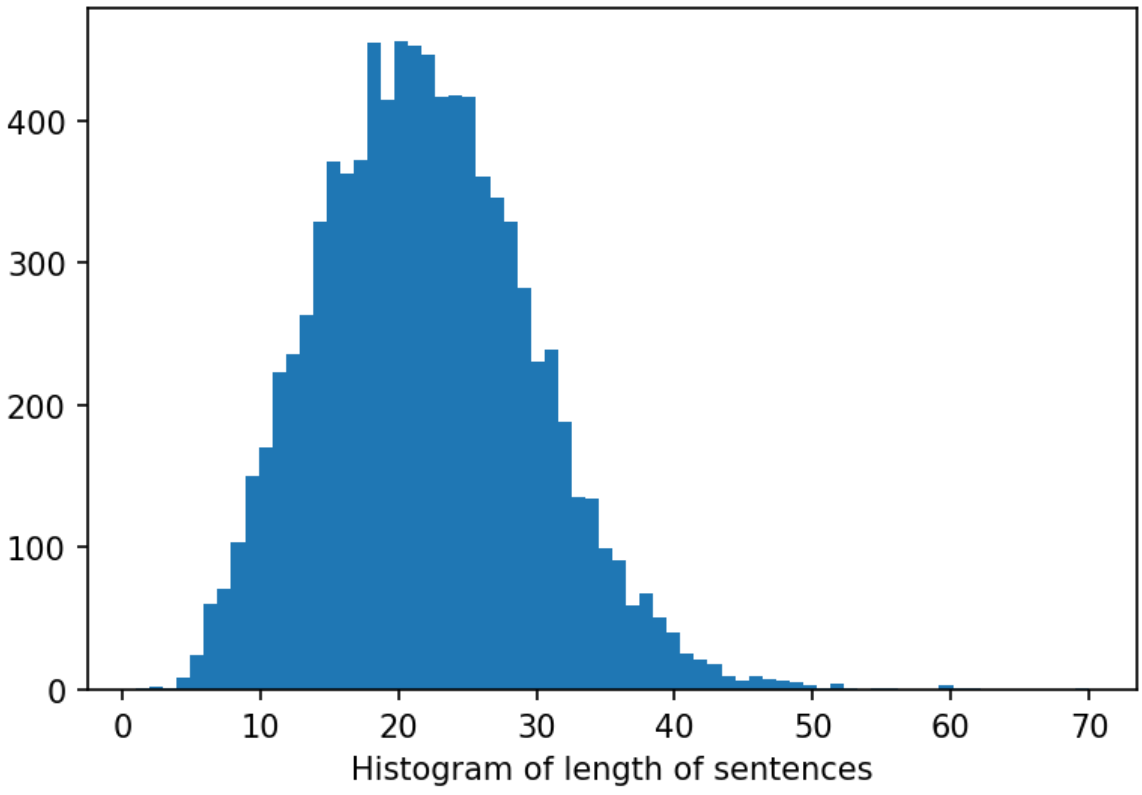
	Label	1	2	3	4	5
0	B-event	II	World	Olympic	Australian	Wilma
0	I-event	War	Open	Games	Cup	II
0	B-geo	U.S.	Iraq	United	Israel	Iran
0	I-geo	States	Korea	Strip	East	of
0	B-gpe	Israeli	Palestinian	Iraqi	Afghan	Russian
0	I-gpe	Serb	States	Korea	d'Ivoire	Republic
0	B-obj	Twitter	GDP	English	Vioxx	Facebook
0	I-obj	Airport	of	Agreement	Trade	Street
0	B-org	U.S.	United	U.N.	Taleban	Hamas
0	I-org	Nations	of	and	Union	House
0	B-per	Mr.	President	Prime	Ms.	General
0	I-per	Bush	Minister	Abbas	Chavez	Obama
0	B-time	Wednesday	Tuesday	Sunday	Saturday	Thursday
0	I-time	of	-	and	,	2003

Érdemes kiemelni pár észrevételt ebben az megjelenítésben: A United States, USA, US kifejezések több osztály esetében is bekerültek a leggyakoribb szavak közé. Ez a későbbiekben problémát jelenthet. Hasonlóan, az “of” viszonyszó gyakran megjelenik egy névelemben, pl 4th of July, United States of America, stb.

Az előző megállapítások nyomán elkezdtem megvizsgálni az egyes szavakat kontextusban is. Találtam érdekességeket az adathalmazban, például a következő esetben téves a “B-time” osztályozás:

	Sentence #	Word	POS	Tag
57298	Sentence: 2594	before	IN	O
57299	Sentence: 2594	his	PRP$	O
57300	Sentence: 2594	arrival	NN	O
57301	Sentence: 2594	,	,	B-time
57302	Sentence: 2594	Mr.	NNP	B-per
57303	Sentence: 2594	Aziz	NNP	I-per
57304	Sentence: 2594	warned	VBD	O

Az ilyen és ehhez hasonló hibák sajnos rontják az elkészített model teljesítményét.
Tovább haladva, megvizsgáltam a korpuszt a mondatok szintjén is, kezdve azok hosszával: A mondatok átlagos hossza 21.85, szórása 7.878 szó. A leghosszabb mondat 70 szavas, ehhez méreteztem a későbbiekben a neurális háló bemenetének hosszát.



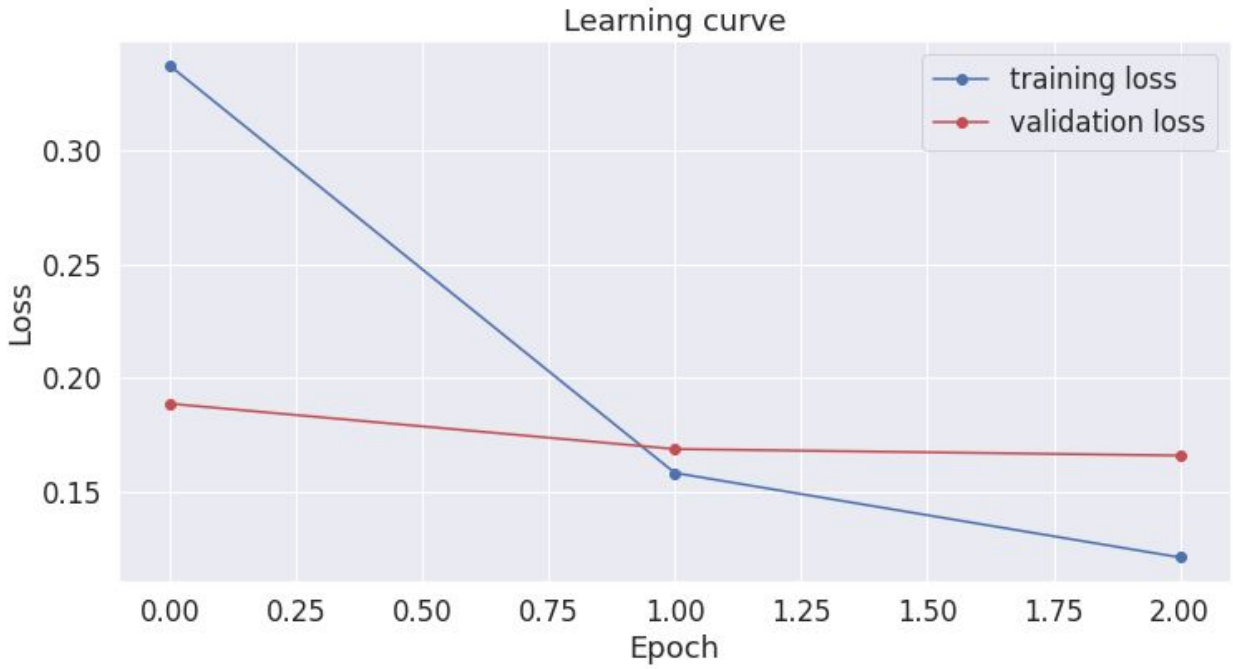
A meglévő megoldás megismerése

Mivel nem dolgoztam még korábban olyan projekten, ahol előtanított BERT neurális hálót használtam volna, és a tanítás implementációja sok időt vett volna igénybe, egy meglévő megoldáshoz folyamodtam, amit a későbbiekben módosítottam. Találtam egy megoldást, amit Tobias Sterbak, egy szabadúszó adattudós készített.
(A megoldás leírását lásd a következő oldalon: https://www.depends-on-the-definition.com/named-entity-recognition-with-bert)

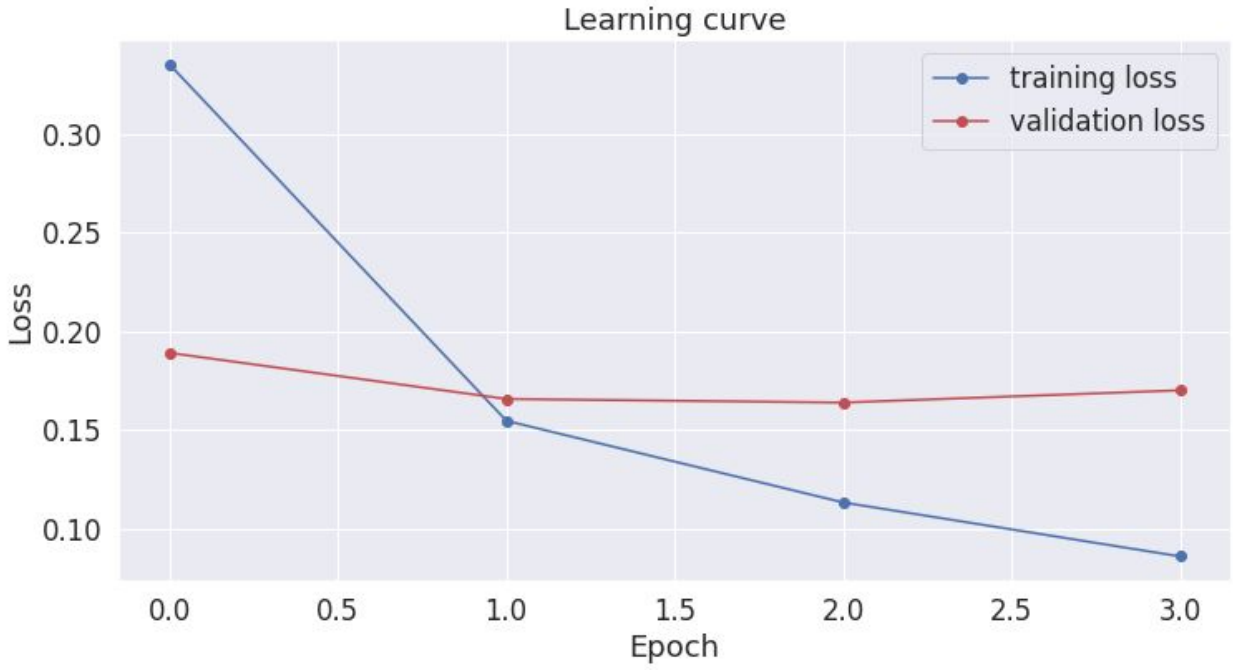
Az itt található előfeldolgozási és tanítási lépéseket először megismételtem, csak a kisebb hibákat javítottam, illetve egy részletesebb kiértékelést adtam hozzá a meglévők mellé.
A tanítás előtt szükséges tokenizálást szintén a transformers csomagban elérhető BERT-specifikus tokenizáló valósítja meg. Ez a tokenizáló a WordPiece tokenizálóra épül, ami elvégzi a szótövezést is. Ezután megtörténik a “padding”, azaz a mondatok kiegészítése megfelelő hosszúságúra. A kiegészítések egy dedikált osztályt, a “PAD” osztályt kapják. A padding után az adathalmazt felosztjuk tanító és validáló halmazra.
Az előtanított modell betöltése és a tanítási paraméterek megadása után elkezdődhet a tanítás.
Egy tanítás a kezdeti paraméterekkel 10 percig tartott a Google Colab rendszerben egy Tesla T4-es GPU-n. Ennek a tanításnak a kimenete a következőképpen nézett ki:

```
Validation loss: 0.16599650796631288
Validation Accuracy: 0.9514913727763163
Epoch: 100%|██████████| 3/3 [05:54<00:00, 118.28s/it]
              precision    recall  f1-score   support

     B-event       0.33      0.12      0.18         8
       B-geo       0.81      0.86      0.84      1062
       B-gpe       0.88      0.84      0.86       400
       B-obj       0.50      0.11      0.17        19
       B-org       0.73      0.68      0.70       641
       B-per       0.83      0.84      0.84       522
      B-time       0.85      0.82      0.83       417
     I-event       1.00      0.14      0.25         7
       I-geo       0.73      0.72      0.73       199
       I-gpe       0.00      0.00      0.00        12
       I-obj       0.00      0.00      0.00         6
       I-org       0.69      0.74      0.72       375
       I-per       0.85      0.93      0.89       695
      I-time       0.69      0.58      0.63       120
           O       0.99      0.99      0.99     17946

    accuracy                           0.95     22429
   macro avg       0.66      0.56      0.58     22429
weighted avg       0.95      0.95      0.95     22429
```

Az “accuracy” metrika sajnos elég naiv, mivel nem veszi figyelembe az egyes osztályok előfordulási gyakoriságát. A “macro avg” sorban található értékek sokkal reprezentatívabbak a nem “other” osztályokra nézve, a későbbiekben pedig az így számolt “f1-score” alapján állapítottam meg, hogy a létrehozott modell jobb vagy rosszabb az eddigieknél.
Az első tanítás során számolt hibaértékeket mutatja a következő grafikon:

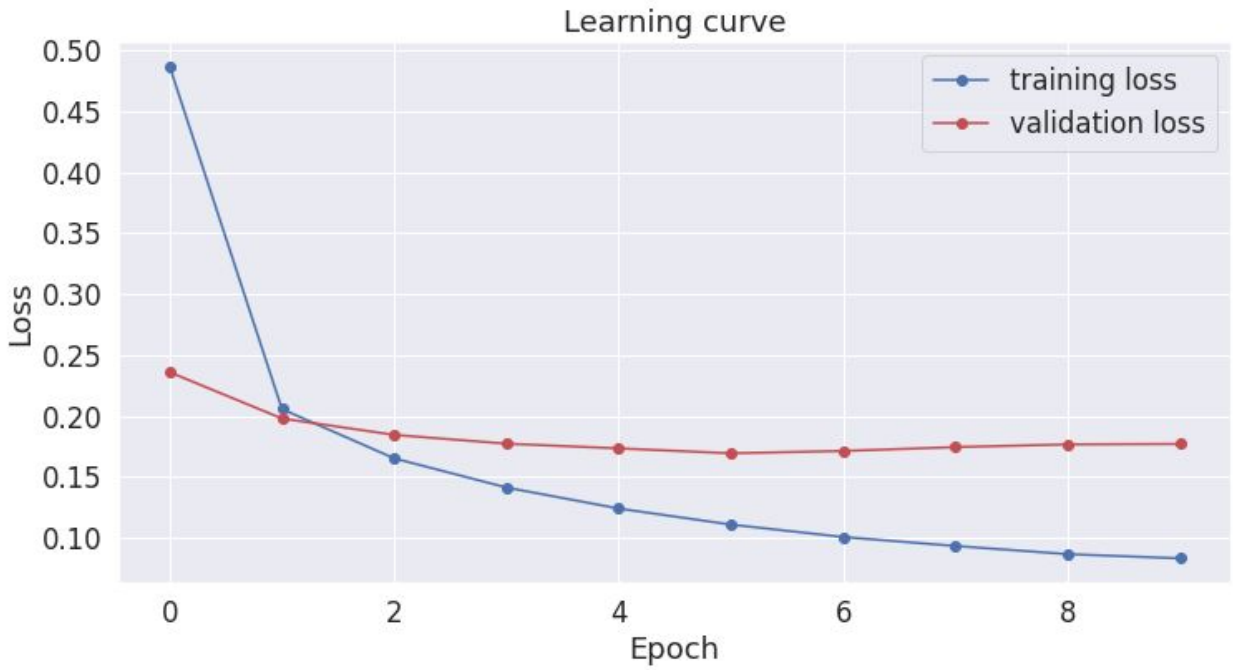


Ami először feltűnt, hogy a kiértékelések száma kevés ahhoz képest, amit az ilyen grafikonokon látni szoktam. Ennek oka a kis epochszám, illetve hogy csak az epochok végén van kiértékelés.
Az epochok számának növelése önmagában nem mindig javít a teljesítményen, hiszen lehetséges, a hogy a neurális háló túltanul.
Ez látható a következő grafikonon:



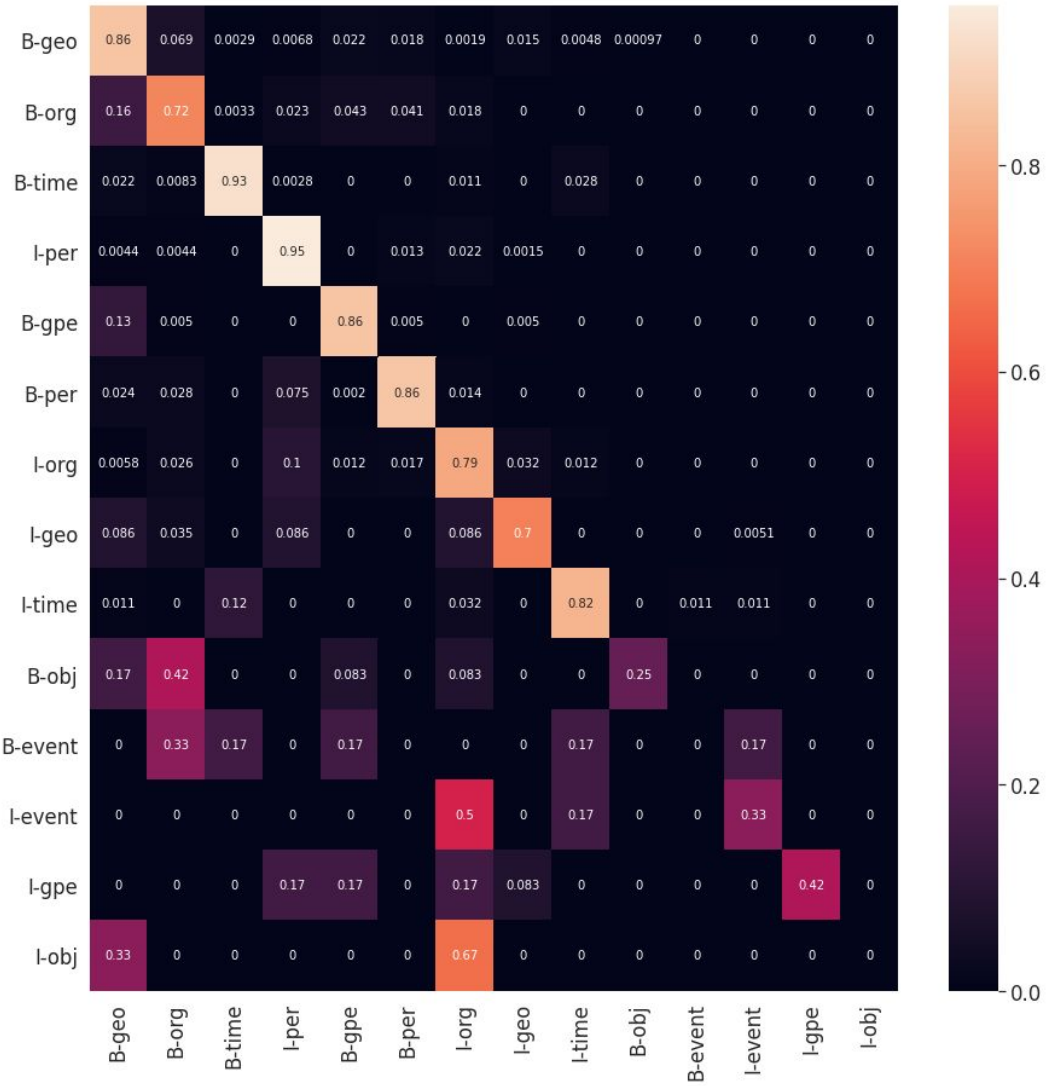
Ekkor a neurális háló a tanuló adathalmaz elemeinek predikcióit egyre pontosabban végzi el, de ennek az eloszlása eltér a validációs halmaztól, így a további tanítás csak ront a modell teljesítményén, ha a validációs halmazra számolunk hibát.

Amivel első körben próbálkoztam, az a validációk számának megnövelése, ugyanis lehetséges, hogy egy köztes kiértékelés jobb teljesítményű modellel történik. Ezt kisebb tanítási rátával és nagyobb epochszámmal értem el, azonban nem hozott lényegi javulást:



Mint látható, az ötödik epochban volt a legkisebb a validációs hiba, ehhez az epochhoz pedig 0.62-es “macro avg f1-score” tartozik, ami kicsit rosszabb mint az eredeti tanításé.

További kiértékelésként előállítottam a predikció confusion mátrixát is, mely a következőképpen néz ki: az egyes sorokban található számok mutatják, hogy mekkora részüket predikálta a model az egyes osztályokba. Ebből kifolyólag a sorösszeg mindig 1. Mint látható, leginkább a mátrix főátlójában találhatók a nagyobb értékek, ami a model egész jó teljesítőképességére utal.



Következő próbálkozásként megpróbáltam egy másik előtanított BERT háló alkalmazását. A transformers csomagban rendelkezésre áll többféle méretben és nyelven a BERT háló. Az “base” méretű háló 109 millió paraméterrel rendelkezik, ami már nem mondható kevésnek. Emellett tanítottam egy “large” méretű hálót is, aminek a paraméterszáma 335 millió. Ennek a tanítása sokkal több időt vett igénybe, és sajnos nem hozta az elvárt jobb eredményt, sőt rontott: 0.55 volt a “macro avg f1-score”. A hosszú tanulási időre való tekintettel elvetettem a további próbálkozásokat ezzel a hálóval.

Az előzőeken kívül voltak még további próbálkozásaim, amelyek a következő paraméterekre irányultak: batch méret változtatása, nagyon tanítóhalmaz és kisebb validációs halmaz alkalmazása, másik optimizer használata (Adam, SGD), a tanulási ráta változtatása, maximális gradiens változtatása, csak az utolsó rétegek tanítása.
Sajnos az előbb felsorolt próbálkozások egyike se tudta érdemben növelni a tanított háló teljesítményét, amelynek a következő két oka van:

- Az alkalmazott előtanított háló sokkal fontosabb, mint az általam hozzáfűzött és tanított pár réteg. Ebből is látszik az előtanítás hasznossága, az iparban is széles körben alkalmazzák a módszert.
- Az alkalmazott paramétereket a BERT szerzői javasolták, tehát ezek valószínűleg a lehető legjobb paraméterek, amiket alkalmazni lehet.

Továbbfejlesztési lehetőségek

A témában lehetséges lenne alkalmazni az újabb state of the art megoldásokat, ugyanis a kettő évvel ezelőtt bemutatott BERT ugyan elterjedt, de nem nyújtja a lehető legjobb teljesítményt. Lehetséges lenne a meglévő adathalmaz bővítése más adathalmazzal, illetve a meglévőben található hibák feltárása és kiszűrése is hasznos lehet.