Multi-label text classification

Protože mám z minulého semestru crawler na články z bbc.com, tak mě napadlo, že ho využiju na stáhnutí článků, na kterých bych multi-label text classification.

Data preprocessing

Stáhnul jsem nějaké články. Dal jsem je dohromady v *merge_data.ipynb.*
Měl jsem hodně různých labelů, tak jsem je pomocí funkce pospojoval.
Např.

Fishing industry → Fish
Fishguard → Fish
Fishing → Fish

Kensington and Chelsea London Borough Council → Kensington
North Kensington → Kensington
South Kensington → Kensington

Ruth Bader Ginsburg → Gin
Gina Miller → Gin

Tato funkce mi vytváří i nesmysly jako *Gina Miller → Gin*, to můžu prozatím ignorovat, protože si budu vybírat labely, kterých je víc jak 100, uvidíme tedy jestli se do tohoto výběru dostanou.
Všiml jsem si, že se mi nespojily labely s policií a koronavirem, to jsem udělal “ručně”.

Nakonec mi vyšli labely:

'US election 2020': 100,
'Social media': 102,
'Social distancing': 102,
'Cardiff': 117,
'Retailing': 129,
'Welsh government': 130,
'India': 132,
'Manchester': 139,
'Self-isolation': 140,
'Personal finance': 153,
'Boris Johnson': 170,
'NHS': 228,
'UK economy': 233,
'Donald Trump': 243,
'Police': 247,
'Brexit': 277,
'China': 296,
'Companies': 313,
'United States': 333,
'Unite': 373,
'Coronavirus': 1898

V tomto výběru nedává smysl akorát label Unite, který vzešel z tohoto:

Unite United States
Unite United Arab Emirates
Unite United Nations
Unite United States Army
Unite Stamp duty in the United Kingdom
Unite United Utilities
Unite United Airlines
Unite United States Senate
Unite United States Postal Service

Methods

Musel jsem pořešit rozdělení dat na testovací a trénovací. Protože když jsem použil funkci *train_test_split(X, y, test_size=0.20, random_state=42)* a pak se podíval na poměr různých labelů, tak se docela liší od požadovaných 20%.

[13.67521368, 14.72868217, 16.54676259, 16.66666667, 16.89189189,
17.81376518, 18.23529412, 18.53035144, 18.62745098, 19.21921922,
19.34156379, 20.38988409, 20.57761733, 21.   , 21.92982456,
23.52941176, 24.28571429, 25.38461538, 25.49019608, 26.60944206]

Udělal jsem vlastní rozdělení pro které vychází poměr labelů:

[19.37984496, 19.42446043, 19.60784314, 19.6969697 , 19.73684211,
19.85559567, 19.93243243, 20.24291498, 20.42042042, 20.44728435,
20.51282051, 20.58823529, 20.58823529, 20.58823529, 21.23287671,
22.14285714, 22.30769231, 22.63374486, 23.17596567, 30.   ]

Moje rozdělení se průměrně více blíží požadovaným 20%. Akorát jedna hodnota vyčnívá s 30%. Ještě by moje metoda na rozdělování šla zlepšit.

Udělal jsem GloVe embedding, jedná se o pretained Glove s embedding layer, potom následuje LSTM a sigmoida.

Ve funkci *model.fit()* používám *validation_split*, to také nemusí být vhodné z předešlého důvodu. Až vylepším funkci na rozdělení testovacích a trénovacích dat, tak jí použiji i na výběr validačních dat.

Další problém je nevyváženost dat, který budu řešit váhami viz odkaz.

Do budoucna porovnám GloVe řešení s modernějším řešením BERT.
Na BERT jsem koukal na články:
https://www.kaggle.com/tanulsingh077/deep-learning-for-nlp-zero-to-transformers-bert
https://atheros.ai/blog/text-classification-with-transformers-in-tensorflow-2