**Fake News Detection Classifier**

Prepared by

Atul Anand Gopalakrishnan

5th Semester, B.Tech(Computer Science)

PES University, Bangalore

Date: 9th August 2019

**Abstract**

The problem statement given addresses the fake news classifier. We build 2 types of classifiers:

* 6-way classifier
* 2-way classifier

The following is covered as a part of this report:

* Acronyms used
* Brief Description of the code
* Idea used
* Accuracy Details
* Citations

**Acronyms used**

* RNN – Recurrent Neural Network
* AWD-LSTM – ASGD Weight Dropped - Long Short Term Memory

**Brief Description of the code**

The code works using the ULMFit algorithm. This is done as follows:

* Extraction of data from the provided dataset: Extract the data from the given git repository and create 3 CSV files for the purpose of convenience, namely – train.csv, test.csv and val.csv from train2.tsv, test2.tsv and val2.tsv respectively.
* Create a language model: A language model is used to predict the next word of a given sentence. This is done using an RNN. The dataset used to create the data bunch is train.csv. The learner is created using the pre-trained model that uses AWD-LSTM architecture and the WikiTexts-2 dataset. The learning rate is graphically computed and used for training the model. The encoder produced is saved.
* Build classifier: A data bunch is again created with the testing dataset. A learner is built with the data bunch and the encoder from the language model. The learning rate is computed graphically and used to fit the data.

**Idea used**

The idea used is the ULMfit algorithm. It is commonly used for problems that address sentiment analysis. The algorithm can be simply explained as follows:

* Create (or, preferred, download a pre-trained) *language model* trained on a large corpus such as Wikipedia (a "language model" is any model that learns to predict the next word of a sentence)
* Fine-tune this language model using your *target corpus*
* Extract the *encoder* from this fine-tuned language model, and pair it with a *classifier*. Then fine-tune this model for the final classification task (in this case, sentiment analysis).

**Accuracy Details**

The highest accuracy for the 2 are as follows:

|  |  |  |
| --- | --- | --- |
| No. of classifications | Test | Validation |
| 6-way | 0.20 | 0.21 |
| 2-way | 0.71 | 0.72 |

**Citations**

The citations for the code are as follows:

* <https://course.fast.ai/videos/?lesson=4>
* <https://nbviewer.jupyter.org/github/fastai/course-v3/blob/master/nbs/dl1/lesson3-imdb.ipynb>
* <https://docs.fast.ai/>
* <https://arxiv.org/abs/1801.06146>
* <https://arxiv.org/abs/1708.02182>
* <https://aclweb.org/anthology/W18-5513>