INF342: Tweet Classification Challenge

Giannis Nikolentzos and Michalis Vazirgiannis

Athens University of Economics and Business

May 2020

1 Description of the Challenge

The goal of this challenge is to study and apply machine learning/data mining techniques to a real-world classification problem. In this classification problem, each sample corresponds to a message posted in Twitter (i.e., tweet). All messages are related to the recent coronavirus outbreak and each message is assigned into some predefined class based on its topic. Besides the textual content of each message, you are also given a graph that models the retweet relationships between users of the platform. Your task is to build a model for predicting the class label of each post. The problem is very related to the well-studied problems of text categorization and node classification. The pipeline that is typically followed to deal with the problem is similar to the one applied in any classification problem; the goal is to learn the parameters of a classifier from a collection of training posts (with known class information) and then to predict the class of unlabeled posts.

The challenge is hosted on Kaggle, a platform for predictive modelling on which companies, organizations and researchers post their data, and statisticians and data miners from all over the world compete to produce the best models. The challenge is available at the following link: `https://www.kaggle.com/c/inf342-datachallenge-2020`. To participate in the challenge, use the following link: `https://www.kaggle.com/t/b71033c6a074445a9ae59e0419749188`. The project's files can be downloaded from this link: `https://www.dropbox.com/sh/lvb4f3ujurj1sm8/AACzFwzGYK33b070Vgk2OTuRa?dl=0`.

2 Dataset Description

As mentioned above, you will evaluate your methods on a dataset of messages posted in Twitter, one of the most popular social media platforms. You are given the following files:

1. **posts.tsv**: it contains information about $16,527$ messages posted in Twitter. For each post, this file contains (1) the post's unique ID, (2) the ID of the user that created the post, and (3) the post's textual content. The tweets are related to the recent coronavirus outbreak and were collected over a period of eight days in April 2020.

2. **users.csv**: the $16,527$ posts were created by $12,640$ users in total. For each one of these users, this file contains (1) his/her number of followers, (2) his/her number of friends (i.e., users that the user follows), and (3) the number of total messages the user has posted.

3. **retweet.edgelist**: a retweet network generated from ~ 8 million messaged posted in April 2020. Nodes correspond to users of the platform and edges to retweet relationships. For example, if user i retweets a post originating from user j, the graph contains a directed edge from j to i. Note

Class	# Train Posts	Class	# Train Posts
0	5,622	1	573
2	2,377	3	1,400
4	1,065	5	180
6	94	7	18
8	122	9	113
10	501	11	324
12	299	13	207
14	326		

Table 1: Size of the 15 classes.

that the edges are weighted since a user may retweet more than one posts originating from another user. The graph consists of 784,898 vertices and 7,401,920 directed edges in total.

4. **train.csv**: it contains 13,221 labeled posts. Each row of the file contains the ID of a post and the class to which it belongs.

5. **test.csv**: this file contains the IDs of 3,306 posts. Each of these posts belongs to one of the 15 possible classes. The final evaluation of your methods will be done on these posts and the goal will be to predict the category to which each post belongs.

With regards the 15 classes, as mentioned above, they correspond to different topics, for instance posts related to countries (e.g., USA) or to fake news (e.g., 5G), etc. The number of samples of each class is illustrated in Table 1. As you can see, the dataset is highly imbalanced.

3 Evaluation

The performance of your models will be assessed using the multi-class logarithmic loss measure. This metric is defined as the negative log-likelihood of the true class labels given a probabilistic classifier's predictions. Specifically, the the multi-class log loss is defined as:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(p_{ij})$$

where N is the number of samples (i.e., posts), C is the number of classes (i.e., the 15 categories), y_{ij} is 1 if sample i belongs to class j and 0 otherwise, and p_{ij} is the predicted probability that sample i belongs to class j.

4 Provided Source Code

You are given two scripts written in Python that will help you get started with the challenge. The first script (`text_baseline.py`) uses features extracted from the textual content of the posts along with a logistic regression classifier, and achieves a log loss score of 1.21120 on the public leaderboard. The second script (`graph_baseline.py`) uses solely graph-based features with a logistic regression classifier for making predictions. This model achieves a log loss score of 1.52708 on the public leaderboard. As part of this challenge, you are asked to write your own code and build your own models to predict the

class to which each post of the test set belongs. You are advised to use both textual information and features extracted from the graph.

5 Useful Python Libraries

In this section, we briefly discuss some packages that can be useful in the project:

- A very powerful machine learning library in Python is `scikit-learn`[1]. It can be used in the pre-processing step (e.g., for feature selection) and in the classification task (a plethora of classification algorithms have been implemented in `scikit-learn`).
- A very popular deep learning library in Python is `Tensorflow`[2]. Since the Keras API is now integrated into TensorFlow 2, you can capitalize on the simple and user-friendly interface of Keras to build and train deep learning models.
- Since you will deal with data represented as a graph, the use of a library for managing and analyzing graphs may be proven important. An example of such a library is the `NetworkX`[3] library of Python that will allow you to create, manipulate and study the structure and several other features of a graph.
- Since you will also deal with textual data, the Natural Language Toolkit (`NLTK`)[4] of Python can also be found useful.

6 Details about the Submission/Presentation of the Project and Grading

Your final evaluation for the project will be based on (1) the presentation you will give (**50**%), (2) on your position on the private leaderboard and the log loss that will be achieved (**20**%), and (3) on your total approach to the problem and the quality of the report (**30**%). As part of the project, you have to submit the following:

- A 3-4 pages report, in which you should describe the approach and the methods that you used in the project. Since this is a real classification task, we are interested to know how you dealt with each part of the pipeline, e.g., how you created your representation, which features did you use, if you applied dimensionality reduction techniques, which classification algorithms did you use and why, the performance of your methods (log loss and training time), approaches that finally didn't work but is interesting to present them, and in general, whatever you think that is interesting to report.
- A directory with the code of your implementation.
- Create a `.zip` file containing the code and the report and submit it to the e-class platform.
- **Deadline: 21/6/2020 23:59**

Presentation: You will also be asked to present the approach you followed. Therefore, you will need to prepare some slides (using ppt or any other tool you like).
Date of presentation: 25/6/2020

[1] `http://scikit-learn.org/`
[2] `https://www.tensorflow.org/`
[3] `http://networkx.github.io/`
[4] `http://www.nltk.org/`