CSC_51054_EP - Machine and Deep Learning
Data Challenge: Sub-Event Detection

November 2024

1 Introduction

The goal of this data challenge is to study and apply machine learning/artificial intelligence techniques to a real-world classification problem. **In this binary classification problem, your mission is to build a model that can accurately predict the occurrence of notable events within specified one-minute intervals of a football match.** You are provided a Twitter dataset [1] centered on the 2010 and 2014 FIFA World Cup tournaments, organized by one-minute long time periods. Each period is annotated with a binary label: 0 if no notable event occurred, or 1 if a significant event—such as a goal, half time, kick-off, full time, penalty, red card, yellow card, or own goal—occurred within that period. For an interval to be labeled as containing an event, it must align closely with the actual event time, without excessive delay.

This setting presents several challenges, from data preprocessing and feature extraction to model selection and hyperparameter tuning. You can employ both traditional machine learning and deep learning techniques to tackle this binary classification task. **The primary evaluation metric will be accuracy, measuring the percentage of correctly predicted labels.** This data challenge is hosted on Kaggle as an in-class competition. To participate, you must have a Kaggle account. If you don't have one, you can create it for free. The URL to register for the competition and gain access to all relevant materials is the following:

https://www.kaggle.com/t/946c29c13d024ffcad34ab0b40e85688

2 Dataset Description

As part of the challenge, you are given the following files:

[1]Meladianos, P., Xypolopoulos, C., Nikolentzos, G., & Vazirgiannis, M. (2018). An optimization approach for sub-event detection and summarization in twitter. In *Advances in Information Retrieval: 40th European Conference on IR Research, ECIR 2018, Grenoble, France, March 26-29, 2018, Proceedings 40* (pp. 481-493). Springer International Publishing.

- train_tweets/*.json: This directory contains all annotated tweets split by their corresponding match. Each file contains tweet data divided into time periods, with each entry labeled as 0 or 1 based on the presence of sub-events.
- eval_tweets/*.json: This directory contains all tweets that need to be annotated
- baseline.py: This script contains two simple baselines, a **Logistic Regression** classifier and a **Dummy Classifier** that always predicts the most frequent class of the training set. You can use the code provided here as a starting point on how to read and process the data and
- logistic_predictions.csv and **dummy_predictions.csv**: sample submission files in the correct format for the two provided baseline classifiers.

3 Task and Evaluation

For each time period in the test set, your model should predict whether a specific sub-event occurred based on the provided tweet data. The evaluation metric for this competition is **accuracy**. Accuracy measures the proportion of correct predictions your model makes, calculated by dividing the number of correct predictions by the total number of predictions. For this binary classification task, the accuracy metric is defined as follows:

$$\text{Accuracy} = \frac{1}{N}\sum_{i=1}^{N} I(\hat{y}_i = y_i)$$

where:
- N is the total number of observations.
- $\hat{y}_i$ is the predicted label for observation i.
- y_i is the actual label for observation i.
- $I(\hat{y}_i = y_i)$ is an indicator function that equals 1 if the prediction matches the actual label, and 0 otherwise.

4 Provided Source Code

You are given two baseline classifiers, written in Python, that will help you get started with the challenge. The script **Dummy Classifier** constantly predicts the most frequent label that appears in the training set. This baseline achieves **0.65234** accuracy in the public Kaggle leaderboard. The classifier is based on **Logistic Regression** that only uses the text of the tweet. The method originally performs a basic preprocessing and then transforms the text using GloVe vectors trained on a Twitter dataset. The accuracy score of this baseline is approximately **0.62890**.

Both baselines ignore the order of the tweets and treat each time period as unrelated to the football match they belong to.

5 Useful Python Libraries

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

- A very powerful machine learning library in Python is scikit-learn[2]. It can be used in the preprocessing step (e.g., for feature selection) and in the classification task (several regression algorithms have been implemented in scikit-learn).
- A very popular deep learning library in Python is PyTorch[3]. The library provides a simple and user-friendly interface to build and train deep learning models.
- Since you will deal with textual data, the Natural Language Toolkit (NLTK)[4] of Python can also be found useful.
- Gensim[5] is a Python library for unsupervised topic modeling and natural language processing, using modern statistical machine learning. The library provides all the necessary tools for learning word and document embeddings. An alternative to it is FastText[6].
- If you do not want to spend a lot of time producing the word embeddings, you can use transfer learning. You can download a pre-trained set of word embeddings on GoogleNews[7], Glove[8] or Numberbatch[9]).
- In case you want to work with the data represented as a graph, the use of a library for managing and analyzing graphs may prove to be important. An example of such a library is the NetworkX[10] library of Python that will allow you to create, manipulate and study the structure and several other features of a graph.

6 Grading Scheme

Each team has to be composed of 2 or 3 students. No larger or smaller teams will be accepted. Group choices need to be submitted to us by **Friday 22nd November at 17:00** at the below link:

[2] http://scikit- learn.org/
[3] https://pytorch.org/
[4] http://www.nltk.org/
[5] https://radimrehurek.com/gensim/
[6] https://fasttext.cc/
[7] https://code.google.com/archive/p/word2vec/
[8] https://nlp.stanford.edu/projects/glove/
[9] https://github.com/commonsense/conceptnet-numberbatch
[10] http://networkx.github.io/

https://forms.gle/G3YQVj8AkX7GHL8p8

Grading will be out of 100 points in total. Each team should deliver:

A submission on the Kaggle competition webpage. (20 points) will be allocated based on raw performance only, provided that the results are reproducible. That is, using only your code, the data provided on the competition page and *any additional resources you are able to reference and demonstrate understanding of*, the jury should be able to train your final model and use it to generate the predictions you submitted for scoring.

A zipped folder on moodle (30 points) including:

1. A folder named ”code” containing all the scripts needed to reproduce your submission.
2. A README file with brief instructions on how to run your code and where it expects the original data files.
3. A report (.pdf file), of max 3 pages, excluding the cover page and references. In addition to your self-contained 3-page report, you can use up to 3 extra pages of the appendix (for extra explanations, algorithms, figures, tables, etc.). Please ensure that both your real name(s) and the name of your Kaggle team appear on the cover page.

The 3-page report should include the following sections (in that order):

- **Section 1: Data Preprocessing and Feature Selection/Extraction (10 points).** Independent of the prediction performance achieved, the jury will reward the research effort done here. You are expected to:
    1. The steps taken to clean and preprocess the data, including tokenization, lowercasing, and handling any potential duplicates.
    2. Explain the motivation and intuition behind each feature. How did you come up with the feature (e.g., are you following the recommendation of a research paper)? What is it intended to capture?
    3. Rigorously report your experiments about the impact of various combinations of features on predictive performance, and, depending on the regressor, how you tackled the task of feature selection.
- **Section 2: Model Choice, Tuning and Comparison (15 points).** Best submissions will:
    1. Compare your model against different classification models (e.g., Neural Network, Random Forest, Graph-based event detection, LLMs, ...).
    2. For each classifier, explain the procedure that was followed to tackle parameter tuning and prevent overfitting.
- Report and code completeness, organization and readability will be worth **5 points**. Best submissions will (1) clearly deliver the solution, providing detailed explanations of each step, (2) provide clear, well-organized and commented code, and (3) refer to research papers. You are free to search for relevant papers and articles and try to incorporate their ideas and approaches into your code and report as long as (a) it is clearly cited

within the report, (b) it is not a direct copy of code and (c) you are able to demonstrate understanding of the content you incorporated.

An oral presentation of your project and the achieved results (50 points) Oral presentations will be scheduled **in the week of the 16th of December**. A schedule on which you can choose presentation slots will be released after the team submissions have been made. We ask you to prepare 10-minute presentations, which will then be followed by questions from the examiners. It is required that all group members are present and take a speaking role during the presentation.

Please note that the **inclusion of any external data sources that directly contain the labels will result in you getting 0 points** on the kaggle submission and being penalised on the report and presentation.

7 Submission Process

Submission files should be in **.csv format**, and contain two columns respectively named "ID" and "EventType". The "ID" column, which is a concatenation of the “MatchID” and “PeriodID” columns, is used to identify each data sample and therefore needs to appear as-is in the results file. The “EventType” column takes 0 or 1 values and is the output of your classifier. Note that two sample submission files are available for download **(logistic_predictions.csv and dummy_predictions.csv)**. You can use it to test that everything works fine.

The competition ends on **Thursday 12th December at 17:00**. This is the deadline for you to submit a compressed file containing your source code and final report, explaining your solutions and discussing the scores you have achieved. Until then, you can submit your solution to Kaggle and get a score at most 5 times per day. There must be one final submission per team.