# PAN at CLEF 2022: Profiling Irony and Stereotype Spreaders on Twitter

Authors:
Jose Arias Moncho
Victoria Beltrán Domínguez

# 1. Introduction

Even though language is the principal method of human communication, every person uses different techniques in order to express themselves and emphasize their intentions. Speaking about literary techniques, irony is used daily by millions of people. This technique consists of saying or doing the opposite of what you mean. If we look at a more aggressive type of irony, we find sarcasm, which uses words, often in a humorous way, to mock someone or something.

In this report, we are going to present and describe the different methodologies proposed in our participation at PAN 2022 at CLEF for the shared task: Profiling Irony and Stereotype Spreaders on Twitter (IROSTEREO).

In order to explain the different implemented approaches, we will first describe the available dataset, as well as the defined partitions. Once the dataset is clearly defined, we will explain the tested models and experiment designs we have tried as well as the obtained results. Lastly, we will specify which of the presented models was proposed in the early bird and in the final submission of the task, as well as the obtained results.

Before continuing, we would like to point, that we have implemented several models: a voting system using the best models, a CNN with the BERT vectors… but in this report, only the ones obtaining the best results will be described, and the others will be left out.

# 2. Dataset

Our aim is to identify Twitter profiles which make a recurrent use of irony or sarcasm. For that, we are given a set of users, where for each user we have a timeline of 200 tweets.

To proceed in this task, two partitions of data are given, training and test. Basic statistic of the number of users in each partition are shown:

| **Number of users** | Train data | Test data |
|---|---|---|
| Ironic | 210 | - |
| Non-ironic | 210 | - |
| **Total** | **420** | **180** |

In order to assure the generalization of the trained model, we have defined a 5-Fold cross validation. Due to the nature of this task, we have made a balanced data splitting for the cross validation, in which we assured that train and validation had the same number of samples for both ironic and non-ironic users. With this in mind, we split users as ironic and non-ironic and then assign in each fold 80% of users to train (half of them ironic, the other half non-ironic) and the 20% left to validation, moving the validation data as each fold was created.

## 3. Experiments

Once the dataset and the partitions are clear, we proceed defining the different defined experiments. Our approach to this problem was based on user representation using two different methods of vectorization:

a. Vectorizers based on statistics.

b. Pretrained BERT.

In the following sections, we will describe the different methods with more in-depth details, followed by our results, obtained by doing the previously defined 5-fold cross validation.

### a. Vectorizers based on statistics

With this approach, the idea was to use vectorizers to represent a user as the counts (or some variant) of tokens in his tweets.

We decided to test two different tweet preprocessing options:

- In the first one, we only changed specific numbers to the token “num” and dates to the token “date”, as we thought that this information would be non-relevant and only add redundant vocabulary tokens.

- In the second one, we also removed every token “#user#”, “#hashtag#” and “#url#” as these tokens had been hidden with these masks and could be considered noise for our user representation.

Our experiments showed that our second preprocessing approach was better, as it cleared irrelevant information. After that, our most relevant results are presented in this table:

| **Vectorizer** | **Classifier** | **Accuracy** |
|---|---|---|
| CountVectorizer | LinearSVC C>=0.1 | 91.66 |
| HashingVectorizer | LinearSVC C=1000 | 91.19 |
| CountVectorizer | SVC C=100000 | 92.38 |

### b. Pretrained BERT

In this second approach, the idea was to transform every tweet into a BERT vector of 768 dense elements.

Once this is done, two different approaches arise:

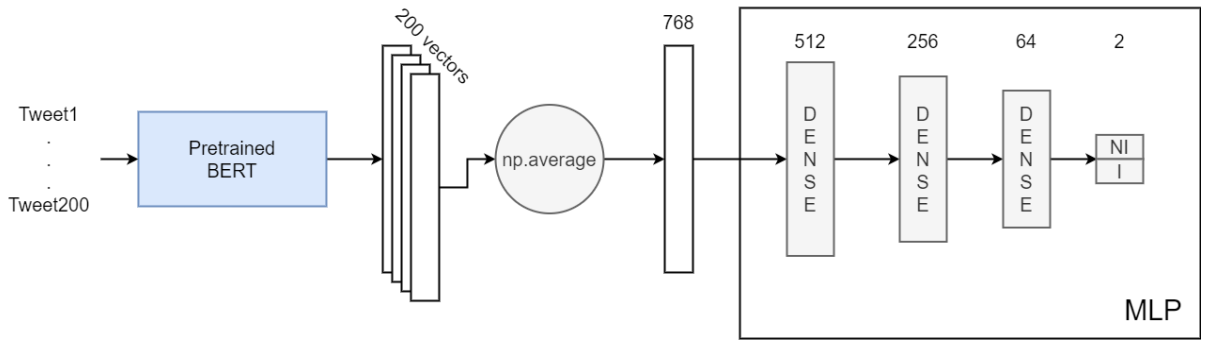
- A concatenation of all of these BERT vectors for every user, resulting in a vector of 153K elements.
- An average of all of these BERT vectors for every user.

The first approach was quite huge to work with, so we decided to apply PCA to 150 and 325 dimensions, as these showed to be the best in our experiments.

| Operation | PCA dimensions | Classifier | Accuracy |
|---|---|---|---|
| Concatenation | 150 | LinearSVC C=1000 | 89.76 |
| Concatenation | 325 | LinearSVC C=1000 | 88.09 |
| **Average** | **-** | **Multilayer Perceptron** | **97.14** |

We also tried obtaining the average vector and applying SVMs, which ended up being a bad idea, as well as doing a concatenation, applying PCA and using a Multilayer Perceptron (MLP), as those results were significantly worse.

As shown above, averaging the obtained embeddings from the pretrained model are our best results. Specifically, our best model (and the one proposed for the **early bird** and **final bird**) was to obtain the average vector of BERT vectors for user tweets and then apply a Multilayer Perceptron following the structure presented here:




Specifically, in the early bird we train this MLP with a 5F-CV and use the 5 resulting models as a voting ensemble. This proved to

produce good results, with an accuracy of 93.33% over the test set.

In contrast, in our final bird, we train this MLP just once with every data as training and predict over the test set, without any evaluation.

## 4. Conclusion

In this report, we have exposed the different models designed in order to detect ironic and non-ironic profiles in twitter. We have tried several methodologies: different vectorizations of users based on statistics, using pretrained models, applying PCA to the resulting results… The one which works best was directly applying a pretrained model (BERT), averaging all the tweets obtained for each user, and passing that information to an MLP.

From all of this work, we conclude that using a pretrained model like BERT, trained with millions of data, can produce a discriminative representation of the content of the tweet that, combined with the MLP, enables to easily classify ironic vs non-ironic profiles.