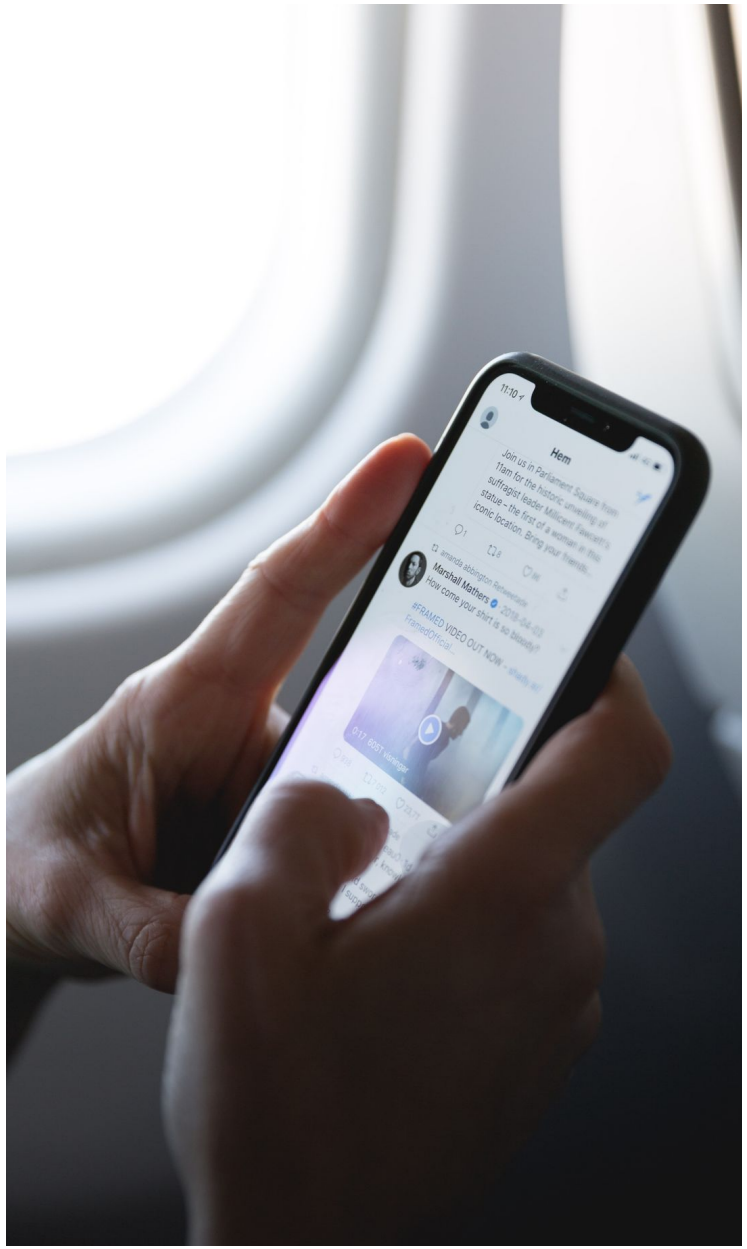
DEBORAH DORE

Sarcas Tweet

Sarcasm classification of italian Tweet

Introduction

The goal of this project is to present a classification model that is able to distinguish between two types of tweets written in Italian : sarcastic and non-sarcastic.



> “A proposal to @Twitter. It would be useful to have an “irony reporting”. A stuff that makes clear to everyone - even and above all to the less ironic - the moment in which you are on Tw in irony mode”

Enrico Letta
Italian Politician

1. Data Collection

Register on the Twitter Developer Portal

Twitter requires the subscription to its Developer Plan in order to be able to download tweets and use them for research purposes

Download Tweet using the Twitter API

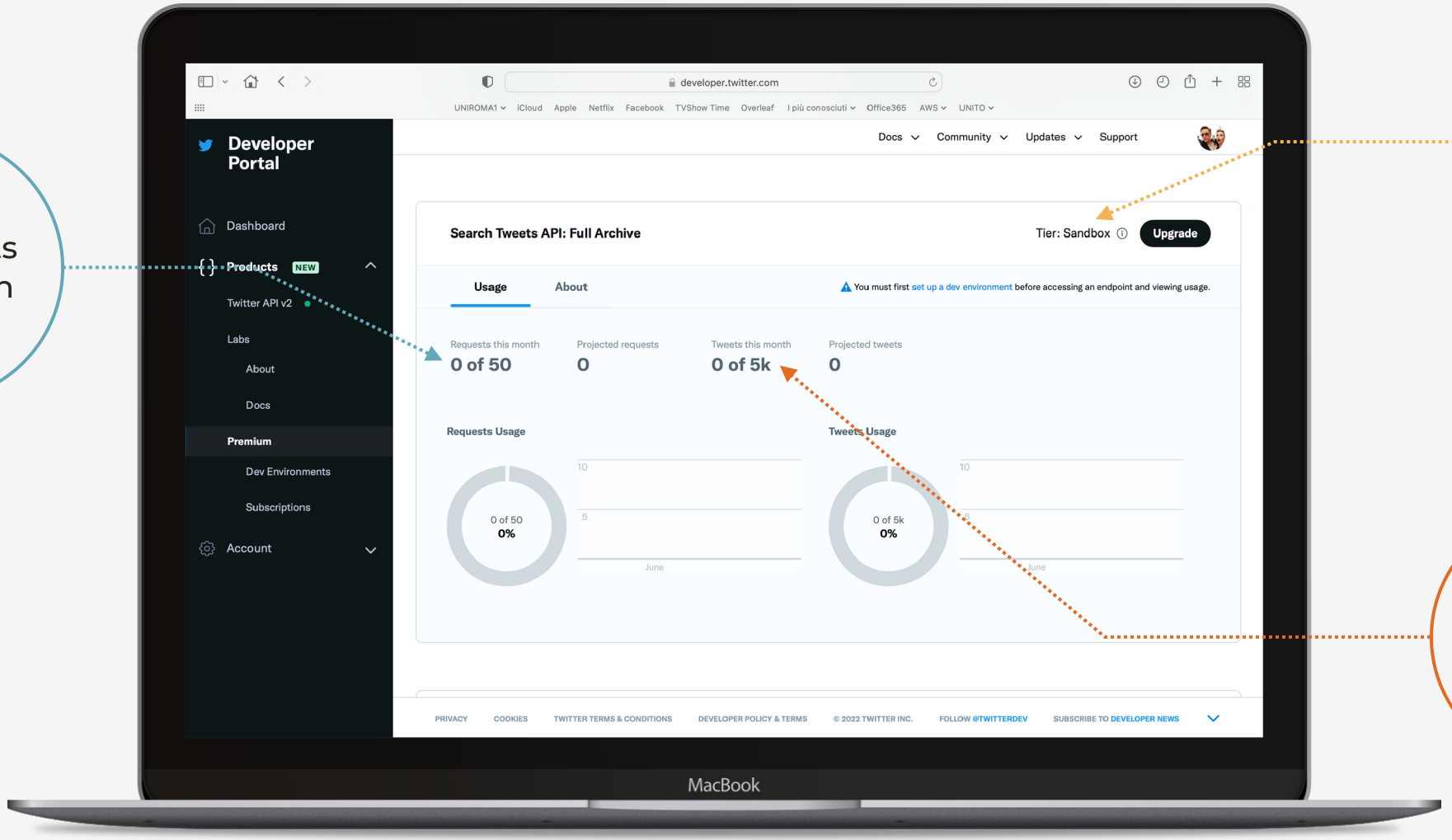
To search for a specific tweet, one must develop a query that will be passed in the request, along with the credentials provided by the Developer portal

Annotate each Tweet using a custom application

Errors can be made even with the use of queries. Therefore it's crucial to check if the tweets were correctly assigned to one of the two classes

Twitter Developer Portal

50 requests per month

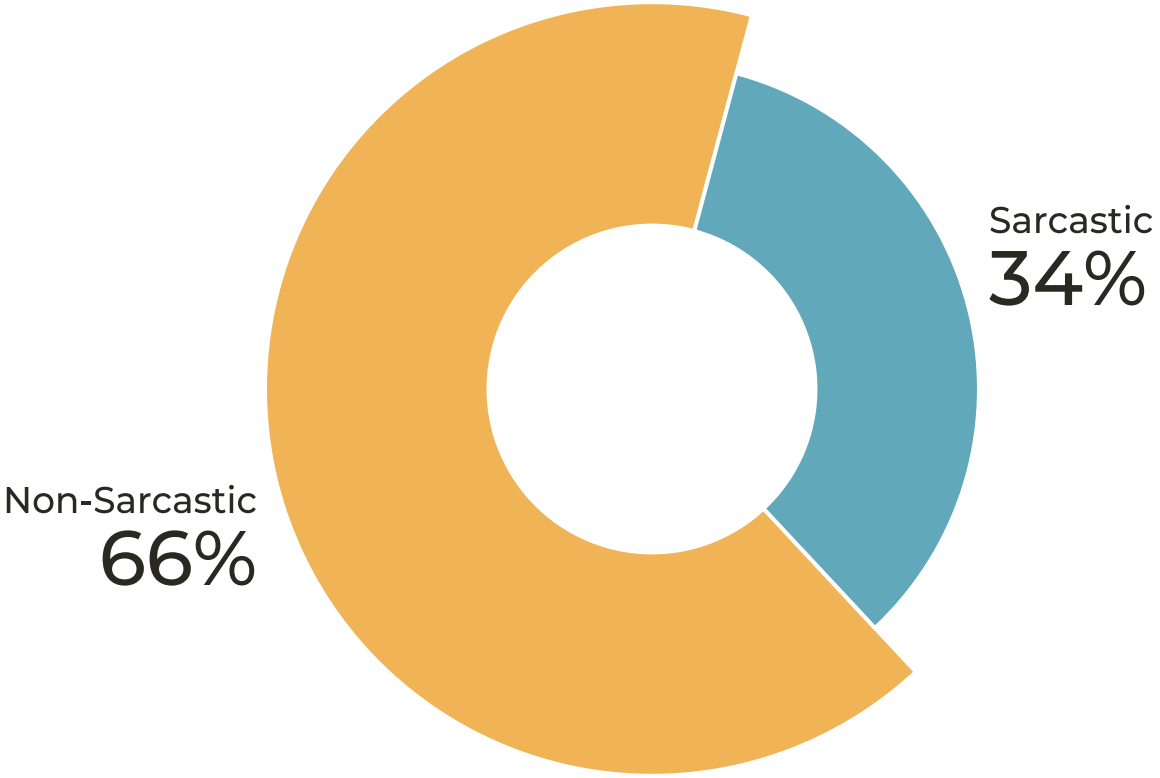


SandBox environment

Maximum of 5k of tweets pulled

Retrieve Tweets

Terms such *as #sarcasmo*, *#ironia*, *ridiamo per non piangere*, *coincidenze io non credo*, and keywords related to politics (a known source of sarcasm) were used to find sarcastic tweets. In the query for non-sarcastic tweets, filtering operators were used to retrieve tweets that didn't contain those terms.

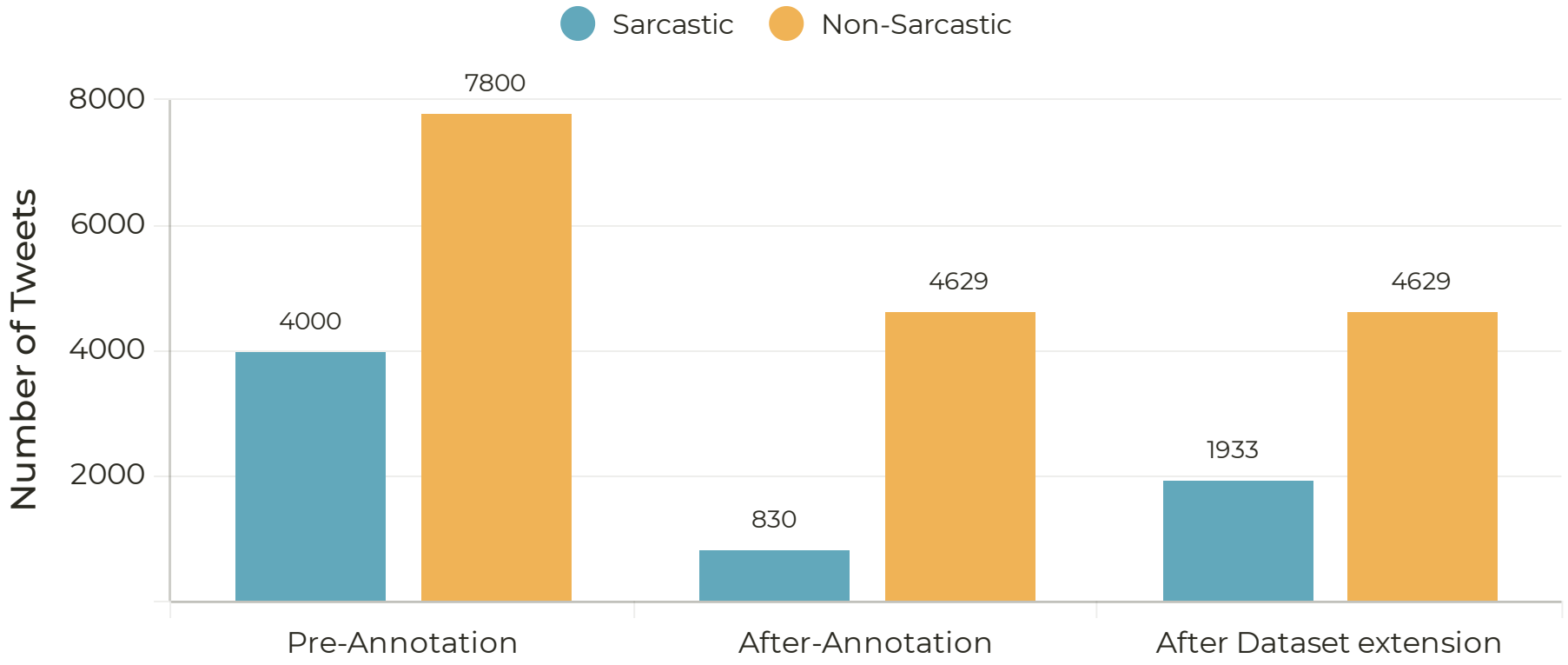


11800

TWEETS

Annotate Tweet

Many tweets that made no sense, or were composed solely of one word, or were exclusively mentions or hashtags were removed during the annotation process.
Numerous tweets that were also wrongly classified were corrected.
The dataset was then extended with the EVALITA2016 Challenge's SENTIPOLIC Dataset.



FINAL RESULT:

6583

TWEETS

2. Data Processing

1 | Removing rows with NaN values

2 | Converting to lowercase

3 | Removing links from text

4 | Removing exceeding symbols

5 | Removing digits

6 | Removing exceeding white spaces

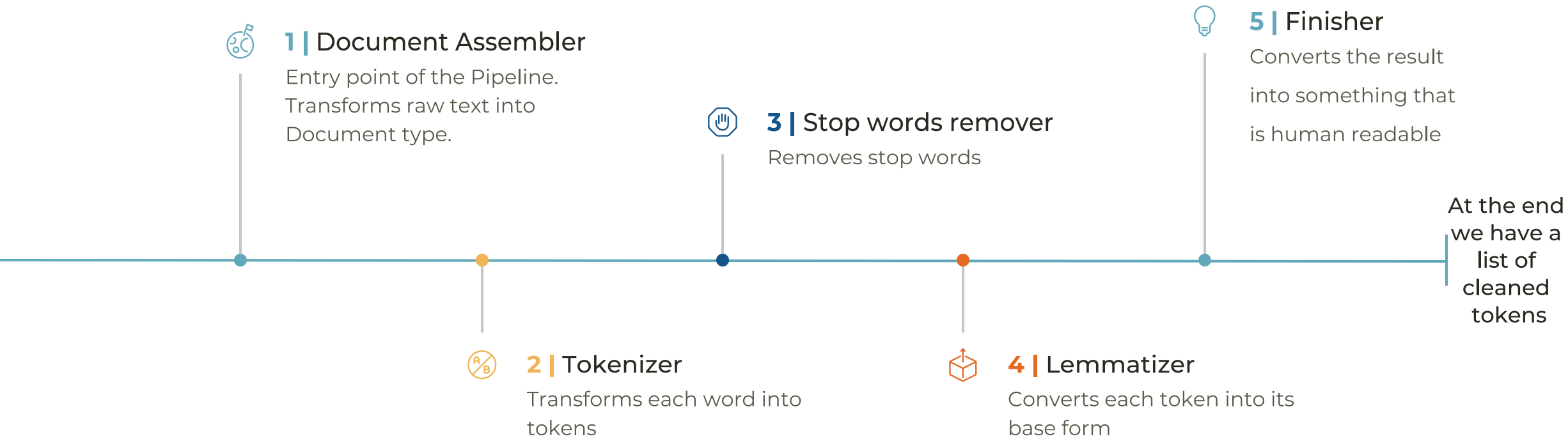
7 | Correcting the spelling

8 | Annotate the text using Ekphrasis

“ <user> user_name </user> che gli vuoi rispondere il vuoto cosmico è meno vuoto della sua testa <hashtag> sarcasmo </hashtag> ”

3. Feature Engineering

With the use of Spark-NLP



4. Model Training: *Word Embedding*

We cannot feed text-like tokens to a model. We need to transform them into something that the model can understand: numbers! Two approaches were used.

Word2Vec

1. Word2Vec models are usually 2-layer neural networks that are pre-trained to reconstruct linguistic contexts of words.
2. It creates a space of corpus where at each word is assigned a vector in the space.
3. Usually words with common context are located close to each other.

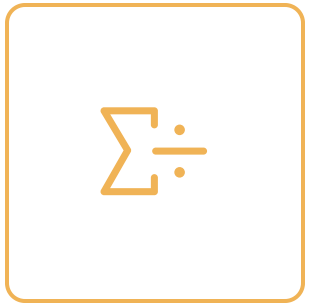
Bert Embeddings

1. BERT stands for Bi-directional Encoder Representations from Transformers.
2. Embeddings in BERT are vectors that encapsulate the meaning of the word
3. Different to Word2Vec, BERT will generate two different vectors for the same word used in two different contexts.

4. Model Training: 3 different models

Supervised learning techniques in close cooperation with cost-sensitive learning methods with the aim of reducing the gap between the majority and minority class

Neural Network + Word2Vec



Convolutional Neural Network + Word2Vec

Logistic Regression + Bert Embeddings

5. Evaluation

We are mostly interested in the Precision and Recall Metrics since the accuracy is not reliable in case of an unbalanced dataset

	Neural Network	Convolutional Neural Network	Logistic Regression
Accuracy	74%	69%	**78%**
Precision on the sarcastic class	**57%**	31%	50%
Precision on the non-sarcastic class	**82%**	81%	80%
Recall on the sarcastic class	**59%**	36%	13%
Recall on the non-sarcastic class	81%	78%	**96%**
F1-Score on the sarcastic class	**58%**	34%	21%
F1-Score on the non sarcastic class	81%	79%	**87%**

Area Under the ROC-Curve and the Precision-Recall Curve

	Neural Network	Convolutional Neural Network	Logistic Regression
AUROC	**0.696**	0.524	0.551
AUPR	**0.504**	0.228	0.392

Upgrades to the Neural Network

Changes

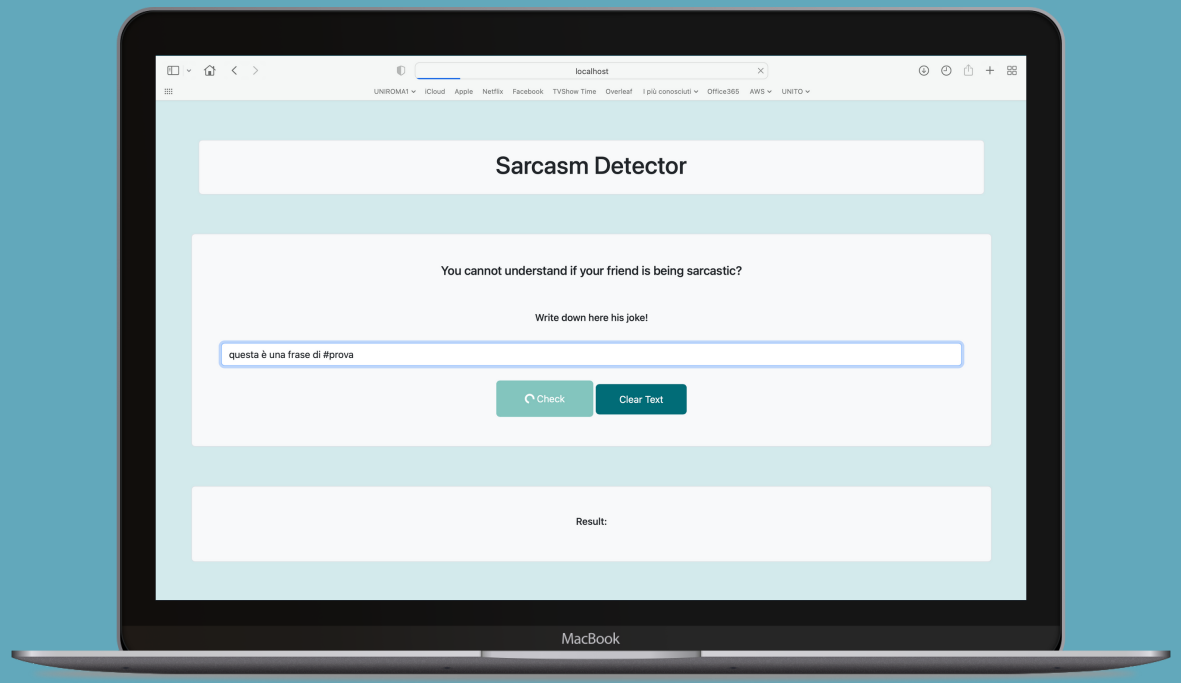
- Increased Complexity
- Increased number of epochs
- Increased learning rate

Results

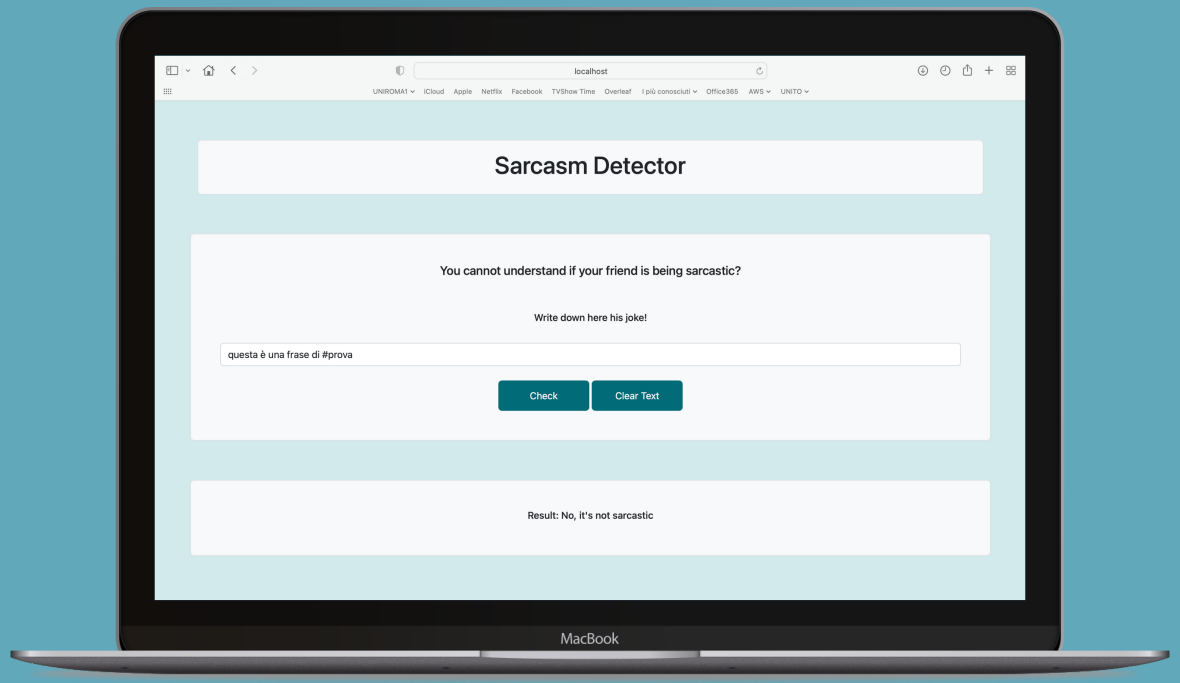
- Increased AUPR
- Increased F1-Score on the sarcastic-class

6. Try it out

Try the best model using a custom designed web app
Pull from the Docker Hub using **docker pull deborahdore/sarcastweet**
Run the container using **docker run -d -p 5000:5000 deborahdore/sarcastweet**
Open browser at localhost:5000



Insert a phrase



Look at the result

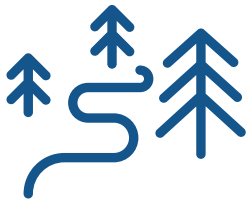
7. Future Work

What can be done to improve the model?

Multiple people as annotators

Cross-Checking on the annotation

Different data processing techniques



Different models like random forest

Add tweets to the dataset

Thank you for your attention