In: the 15th International Conference on Semantics, Knowledge and Grids, Guangzhou, China, 2019

Discovering Political Polarization on Social Media: A Case Study

Loris Belcastro[#], Riccardo Cantini[#], Fabrizio Marozzo[#], Domenico Talia[#], Paolo Trunfio[#]

[#]*DIMES, University of Calabria*
Rende (CS), Italy
[#]{lbelcastro, rcantini, fmarozzo, talia, trunfio}@dimes.unical.it

***Abstract*— Social media analysis is a fast growing research area aimed at extracting useful information from social media. This paper presents a methodology aimed at discovering the behavior of social media users during election campaigns characterized by the competition of political parties. The methodology analyzes the posts published by social media users through an automatic incremental procedure based on feed-forward neural networks. Specifically, starting from a minimum amount of classification rules (a small subset of the hashtags that are notoriously in favor of specific factions), the methodology iteratively increases the inferred knowledge by generating new classification rules. These rules are then used to determine the polarization of social media users towards a party. The proposed methodology has been applied on a case study that analyze the polarization of a large number of Twitter users during the 2018 Italian general election. The achieved results are very close to the real ones and are significantly more accurate than the average of the opinion polls, revealing the high accuracy and effectiveness of the proposed approach.**

I. Introduction

Every person connected to the Internet leaves many tracks by interacting with various devices and platforms, from which his/her interests, opinions, consumption behavior or contacts can be recognized. All this leads to large amounts of data, referred as Big Data, which are also characterized by the complexity, by the variety, and by the velocity of data that can be collected and processed [1]. Thanks to their characteristics, Big Data are intrinsically suited to a very large set of applications aimed at understanding how information spreads within a network, or analyzing user behavior, through for example trajectory mining techniques [2], as well as their mood or opinion in relation to a topic or an event of interest. These applications pertain the field of sentiment analysis, in which different natural language processing techniques converge, by virtue of the predominantly textual form of data. This work focuses on Big Social Data [3], the subset of Big Data that comes from the interaction of users with social media, such as Facebook, Twitter, Instagram, YouTube and others, whose use has become part of everyday life. The analysis of Big Social Data, which belongs to the field of social media analysis, is aimed at studying the interactions of certain users on social media in order to outline a profile describing their salient features from a psychological and behavioral point of view. This allows to discover the public opinion in a community of users, modeling precisely their perception of facts, events and public decisions [4].

This work deals with the use of social data, in particular those coming from Twitter, to determine the polarization degree of public opinion in relation to a series of opposing factions during a political event of interest. In particular, a new incremental methodology based on neural networks is proposed, aimed at estimating the polarization of public opinion during a political event, which can be directed towards a particular faction or a candidate, in the case of an election, or towards a binary choice in the case of a referendum.

The proposed methodology has been applied to a case study that analyzes the polarization of a large number of Twitter users during the 2018 Italian general election. In particular, Twitter users have been classified using the polarization rules extracted from our methodology and the results have been compared with opinion polls collected before voting and with the actual results obtained after the vote. The achieved results are very close to the real ones and are significantly more accurate than the average of the opinion polls, revealing the high accuracy and effectiveness of the proposed approach.

We experimentally evaluated the accuracy of our methodology through different statistical indexes, comparing the obtained results with the latest opinion polls published before the elections. In particular, considering the four parties that received the highest number of votes (M5S, PD, LEGA, FI), our methodology achieved a mean average error (MAE) of 1.13 percentage points and a determination coefficient R^2 very close to 1. Instead, opinion polls achieved a MAE of 3.74 percentage points and a value of R^2 of 0.72, which confirm the ability of the proposed methodology to forecast election results.

The remainder of the paper is organized as follows. Section II discusses related work. Section III introduces the neural networks. Section IV describes the proposed methodology. Section VI presents the case study. Finally, Section VII concludes the paper.

II. Related work

In recent years, social media analysis is arousing great interest in the scientific community. Several opinion mining techniques have been developed for capturing the mood of social media users and their opinions related to a specific topic of interest. Several researches are working on the design and implementation algorithms for measuring public opinion and predict election results by classifying social media posts. Existing techniques can be grouped into four main approaches [5]:

1) *Opinion-based*. This approach includes text mining techniques for understanding: $i)$ the feeling of users (e.g., *positive* or *negative*) [6], or also *neutral* [7]; $ii)$ the polarization of users, such as *pro* and *cons* [8], typically opposed in the political field [9].
2) *Volume-based*. This approach aims at predicting the elected candidate or winning party by measuring the volume of posts related to them. For example, some studies found an interesting correlation between the volumes of retweets, mentions and voting percentages recorded during the elections [10].
3) *Opinion and volume-based*. It combines the two approaches described above. For instance, Jahanbakhsh et al. [11] combined opinion-based classes and volume-based measures, while Wong et al. [12] used sentiment-based-classes and retweets information.
4) *Emoji-based*. This approach exploits emojis for classify posts. For example, Chin et al. [13] classified the main emojis according to the emotion they represent; then the mood related to the post is determined based on the first emoji in it.

According to the level of automatism that characterizes the classification process, such techniques can be also divided into two macro-categories: *manual* and *semi-automatic* (or *dynamic*).

In *manual techniques* the classification model is manually defined by domain experts. As an example, Graham et al. [14] performed an hand-coded content analysis for understanding how British and Dutch parliamentary candidates used Twitter during the 2010 general elections.

Semi-automatic techniques exploit text mining and sentiment analysis without the need of an extensive knowledge of the application domain. For example, El Alaoui et al. [5] proposed an adaptive sentiment analysis approach for extracting user opinions about political events. Specifically, the proposed approach classifies the posts by exploiting a series of word polarity dictionaries built from a selected set of hashtags related to a topic of interest.

Marozzo and Bessi [15] presented a methodology aimed at discovering the behavior of social media users and how news sites are used during political campaigns characterized by the rivalry of different factions. The idea behind this technique is to use the keywords inside a tweet to classify it by calculating the degree of polarity.

Oikonomou et al. [16] used a Naïve Bayes classifier with text mining techniques given by TextBlob, a Python library which provides an API for Natural language processing (NLP), to predict the outcome of USA presidential elections in three states of interest (i.e., Florida, Ohio and North Carolina).

Jaidka et al. [17] compared three different methods (i.e., volumetric, sentiment and social media analysis) in order to predict the outcome of the elections from Twitter posts in three Asian countries: Malaysia, India, and Pakistan.

Olorunnimbe et al. [18] presented an incremental learning method based on multiple independent models for predicting the political orientation of users over time. Starting from an annotated corpus produced using domain-specific knowledge, the proposed method exploits an incremental learning approach for training a probabilistic classification model based on Naïve Bayes.

This work proposes a new opinion and volume-based technique for estimating the polarization of public sentiment during a political event of interest. It is an automatic annotation methodology that aims at decreasing the amount of domain knowledge needed for the analysis process. In particular, we exploit an automatic and incremental procedure that, starting from a minimum amount of classification rules (i.e., a small subset of the hashtags that are notoriously in favor of specific factions), iteratively increases the inferred knowledge by generating new classification rules. To do this, the proposed methodology uses a series of feed-forward neural network models, which are able to gradually extract classification rules.

III. Neural networks

Neural networks are computing models which includes a set of techniques aimed at realizing intelligent systems capable of acting rationally within an environment, pursuing a specific objective. Normally they act guided by a specific utility function and use machine learning techniques to induce new knowledge, improving performance through experience. Deep learning, a sub-category of machine learning, creates multi-level learning models, characterized by the automatism of the feature extraction phase. These models [19] are structured on different levels of representation, corresponding to hierarchies of characteristics of concepts, where high-level concepts are defined on the basis of low-level ones learned previously: this allows the modeling of very complex concepts, which gives an excellent capacity of abstraction and generalization. These systems are also characterized by a high degree of flexibility and excellent performance in many application areas, but the models they generate are not easy to understand. This is related to the nature of the models themselves, which use a very large number of parameters.

The main objective of the applications based on neural networks is the development of mathematical algorithms that allow networks to learn by imitating information processing and the acquisition of new knowledge by the human brain. They derive from biological systems the main characteristics of information processing, such as non-linearity, high parallelism, noise robustness, fault and error tolerance, learning, and the ability to generalize. Therefore, such models have great flexibility as they are able to adapt to a wide range of situations and accurately approximate highly dimensional functions, deriving from different application fields, such as image recognition, natural language processing, sentiment analysis or synthetic data generation.

There are various types of neural networks, each with peculiar characteristics, which can also be combined together to obtain complex models. In general, they can be grouped into two main classes:

- *Feed-forward networks*, which can be represented as acyclic direct graphs, including classical architectures such as the multilayer perceptron (MLP) on which the developed methodology is based, or the convolutional neural networks (CNNs) whose architecture is inspired by organization and functioning of the animal visual cortex and is very scalable as the complexity of the input increases.
- *Recurrent networks*, based on the awareness that human reasoning does not start from scratch every time, but thoughts are persistent and past experience is an integral part of the reasoning itself: this concept is totally absent in classical neural networks. Therefore recurrent neural networks present an architecture that allows the presence of cycles in order to make persistence of information possible.

A. Multilayer perceptron

The perceptron [20] represents the simplest form of neuron and consists of a single unit with adjustable weights, $w_j, j = 1, 2, ..., n$. Given an input feature vector $x = (x_1, x_2, ..., x_n)$, the input of neuron is:

$$v = \sum_{j=1}^{n} w_j x_j$$

The perceptron output y is determined by an activation function, represented by the Heaviside step in the classical case, applied to the sum between a threshold term, called bias and the weighted sum in input. There are other activation functions, usually preferred to the step for its non-derivability problems, such as the sigmoid, useful for classification and logistic regression, the ReLu, excellent for maintaining the non-linearity properties, the tanh, which allows output values in the range $[-1, 1]$, or softmax, used for multi-class classification by virtue of its probabilistic properties. The structure of the classical perceptron is showed in Figure 1.

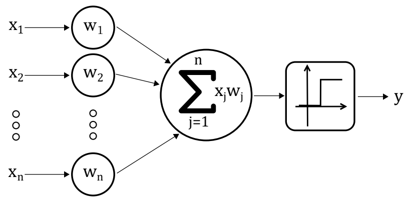

Fig. 1. Structure of a perceptron.

The linear equation $\sum_{j=1}^{n} w_j x_j - b = 0$ defines the decision boundary, a hyperplane of separation in an n-dimensional input space. The perceptron can be trained on a set of examples using a particular learning rule: its weights and bias are changed in proportion to the error, considering the difference between actual and target output, for each example of training. If the considered classes are linearly separable, it is possible to derive a learning rule for the perceptron that produces an optimal vector of weights in a finite number of iteration and independently from the initial values of the weights. In this case in fact, there will exist a linear hyperplane capable of dividing the classes into distinct half-spaces. To deal with non-linearly separable problems, some levels of neurons are introduced between the input level and the output level, which lead to an architecture called multi-layer perceptron. Since these intermediate levels do not interact with the external environment they are called hidden layers. The addition of intermediate levels allows the model to handle non-linear classification problems. These networks are trained through the back propagation algorithm, which is used to calculate the gradient, and Adaptive Moment Estimation (ADAM) [21], which is an optimization algorithm for calculating adaptive learning rates for each parameter, optimizing the learning process.

IV. Proposed Methodology

This work proposes a new methodology for estimating the polarization of public opinion during a political event of interest. Given a political event $\mathcal{P}$, the proposed methodology consists of six steps:

1) Definition of the factions F and collection of the keywords K associated to $\mathcal{P}$.
2) Collection of posts that contain one or more keywords in K.
3) Pre-processing of posts for creating a clean dataset P.
4) Post classification by executing an incremental algorithm based on neural networks on dataset P.
5) User polarization.
6) Visualization of results.

For the sake of clarity, Table I reports the meaning of the symbols used to describe the different steps.

Symbol	Meaning
$F = \{f_1, f_2, ..., f_n\}$	Factions
$K = K_{context} \cup K_F^{\oplus}$	Context keywords and positive faction keywords
$K_F^{\oplus} = K_{f1}^{\oplus} \cup ... \cup K_{fn}^{\oplus}$	Positive keywords grouped by factions
P	All the posts in input
C^i	Classified posts at the i-th iteration
U^i	Unclassified posts at the i-th iteration
M^i	Classification model generated at the i-th iteration
th	Threshold for the probability that a post the supports a faction
eps	Minimum % increment of classified posts at each iteration
max_{iters}	Maximum number of iterations

TABLE I

Meaning of the most important symbols used in the proposed methodology.

A. Definition of the factions F and collection of the keywords K associated to $\mathcal{P}$

A political event $\mathcal{P}$ is characterized by the rivalry of different factions $F = \{f_1, f_2, ..., f_n\}$. Examples of political events and relative factions are: $i)$ municipal election, in which a faction supports a mayor candidate; $ii)$ political election, in which a faction supports a party; $iii)$ presidential election, in which a faction supports a presidential candidate. In this step, we collect the main keywords K used by social media users to write posts associated to $\mathcal{P}$. Such keywords can be divided in those of *context* and those in *favor* of a given faction, i.e., $K = K_{context} \cup K_F^{\oplus}$. Specifically:

- $K_{context}$ contains generic keywords that can be associated to $\mathcal{P}$ without referring to any specific factions in F.
- $K_F^{\oplus} = K_{f1}^{\oplus} \cup ... \cup K_{fn}^{\oplus}$, where $K_{fi}^{\oplus}$ contains the keywords used for supporting $f_i \in F$ (positive faction keywords).

This step requires a minimal knowledge of the domain, that means it can be easily automated. In fact, in most political events, keywords used for supporting a specific faction usually match some fixed patterns, such as the form "#vote + (candidate / faction / yes / no)". In posts gathered from Twitter (tweets), such patterns can be searched in hashtags or words.

B. Collection of posts

In most cases, public APIs provided by social media platforms permit to download all the posts containing one or more *keywords* in K. As an example, Twitter APIs allows for searching and downloading all the tweets containing specific hashtags or words. Posts are not collected in real time, but downloaded a given time after their publication (e.g., 24 hours). In this way, we are able to get some statistics related to the popularity of a post, such as: $i)$ number of *shares*, which indicates how many users shared a post with their friends; $ii)$ number of *likes*, which indicates how many users found a post useful. Each collected post contains at least one key in K, but it may have also other keywords, namely *co-keywords*, that can be exploited for understanding the terms used to support the voting intentions. Since data collection is usually a continuous process, new keywords can be discovered and integrated to the collection K.

C. Pre-processing of posts

During this phase, the posts collected are pre-processed for making them suitable for the analysis. In particular, they are filtered and modified so as to:

- remove duplicates;
- normalize all the keywords by transforming them in lowercase and replacing accented characters with regular ones (e.g., IOVOTOSI or iovotosí $\rightarrow$ iovotosi);
- stem words in text for allowing matches with declined forms (e.g., vote or votes or voted $\rightarrow$ vot);
- remove stop words using preset lists;
- increase the number of keywords considered by selecting the top N words;
- improve data representativeness by filtering out all the posts having a language different from the one spoken in the nation hosting the considered political event.

D. Post classification

The algorithm used for classifying the posts consists of several steps, as described in the following.

The input is composed of a set of posts P that have been collected and filtered as described in Sections IV-B and IV-C, a set of faction keywords K_F, the maximum number of iterations max_{iters}, the minimum increment of the classified posts eps at each iteration, and a threshold th. The output is a dictionary of classified posts that have been assigned to a faction.

As preliminary iteration, the algorithm initializes an empty set C^0 for storing the classified posts and builds a classification model M^0 based on the faction keywords K_F. The model defines a set of rules for calculating a binary vector v_b, where $v_b[i]$ is 1 if p contains at least one hashtag from K_{fi}, 0 otherwise.

The algorithm iterates on each post p in P performing the following operations:

- classifies p using M^0, which produces a vector v_b;
- if the p is in favor of a single faction f then the faction f is found, the pair $\langle p, f \rangle$ is added to dictionary D_P, and p is added in C^0.

At the end of iteration 0, the set of unclassified posts (U^0) is calculated as the difference between P and C^0.

The second part of the algorithm performs at most max_{iters} iterations. Specifically, at the i-th iteration, the following operations are performed:

- it initializes an empty set C^i for storing the classified posts;
- it builds a classification model M^i by training a neural network using the classified posts from previous iterations ($C^0 \cup \ldots \cup C^{i-1}$);
- for each unclassified post at the previous iteration U^{i-1}, the algorithm classifies p using M^i, which produces a vector of probabilities v_p, where $v_p[i]$ is the probability that p supports f_i; if the maximum value of v_p is greater than the given threshold th then the faction f is found, the pair $\langle p, f \rangle$ is added to dictionary D_P, and p is added in C^i.
- it calculates the set of unclassified posts U^i as the difference between the set of unclassified posts at the previous iteration U^{i-1} and the set of classified posts C^i;
- if the ratio between the size of C^i and the size of U^{i-1} is lower than eps or greater that $1 - eps$, then it stops iterating.

Finally, the algorithm returns the dictionary D_P containing all the posts classified at the various iterations and the faction they have been assigned to.

Figure 2 shows how the post classification algorithm works starting from a set of posts P. At the iteration 0, the classification model M^0 is created using the positive faction keywords K_F. This model is used to classify P, which generates two subsets for classified (C^0) and unclassified posts (U^0) respectively. At iteration 1, a new model M^1 is trained using

C^0 and is used to classify the unclassified posts generated at the previous iteration (U^0). The classification process splits U^0 in two new subsets: C^1 for classified posts and U^1 for unclassified ones. The process is repeated in subsequent iterations until the ratio between the size of C^i and the size of U^{i-1} is lower than eps or greater that $1-eps$. At the end, the whole set of classified posts is obtained as the union of the C^i produced at each iteration, while the remaining posts (U^n) are classified as neutrals. For the sake of simplicity, Figure 2 illustrates an example that terminates after only three iterations ($n = 3$).

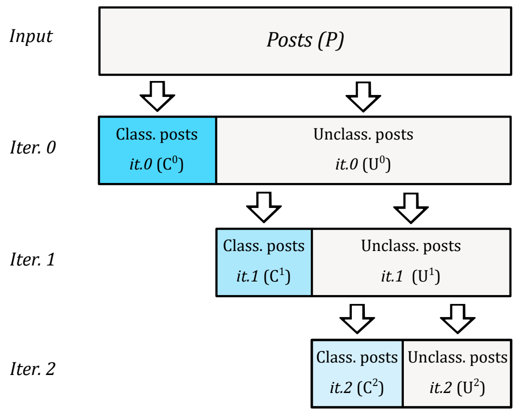

Fig. 2. Example of use of post classification algorithm terminating in three iterations.

Table II describes a usage example of the post classification algorithm on tweets of 2016 U.S. presidential election, characterized by the rivalry between two candidates: *Donald Trump* and *Hillary Clinton*. The input of the algorithm is composed of a set of tweets regarding the political event and a set of faction keywords K_F:

- K_{Trump}={#voteTrump,#MAGA,#makeamerikagreatagain}
- $K_{Clinton}$={#imwithher,#strongertogether,#voteHillary}

At iteration 0, K_F is used to generate M^0, which allows to classify 4 tweets. At iteration 1, classified tweets are used to train M^1. This model generates some rules, such as:

- since Donald Trump has been accused of sexual assault by some women, tweets with keywords *#sex #woman* are classified in favor of Clinton;
- similarly, since Hillary Clinton contravenes the federal laws by using personal email account for government business, tweets with keywords *#email #hillary* are classified in favor of Trump.

At iteration 2, the algorithm learned some new classification rules about immigration, a topic on which the two candidates had an opposite position.

E. User polarization

The algorithm used for estimating user polarization in relation to the considered factions is discussed in the following. The input is composed of a dictionary D_P of classified posts (i.e., output of the post classification algorithm), a filtering function $filter$ and its parameters par_f, and a polarization function $polarize$ and its parameters par_p. The output is a score vector (v_s) containing the global polarization score for each faction, calculated aggregating the scores of all users.

As first step, the classified posts are aggregated by users to produce a dictionary D_U, which contains the list of classified posts C_u for each user u. Let D_S be a dictionary used to store for each user u the score vector v_s^u, which contains his/her polarization percentages towards each faction under analysis. The algorithm iterates on each pair $\langle u, C_u \rangle$ in D_U performing the following operations:

- Filters out all the pairs that do not match the criteria defined by the function $filter$ (e.g., it can skip users having a number of classified posts below a given threshold).
- Using the classified posts C_u, computes v_s^u, a vector containing the score of user u for each faction. The score vector is calculated by using the function $polarize$.
- Adds the pair $\langle u, v_s \rangle$ to D_S.

Then the algorithm computes the score vector v_s by iterating on each pair $\langle u, v_s^u \rangle$ in D_S and adding up the different v_s^u. At the end, the algorithm returns the final score vector, obtained from v_s by normalizing it with respect to its sum.

In our case studies the user polarization algorithm has been configured using the functions and parameters described in the following. First of all, from the list C_u of all the classified posts published by u, we determined his/her favorite faction fav^u as $argmax(C_u)$, the number of posts published in favor of this faction p_{fav}^u as $max(C_u)$, and the total number of tweets posted s^u as $sum(C_u)$.

For each user u, two aspects have been considered by the $filter$ function:

1) $s^u > 5$, which means that u has posted at least 5 tweets during the weeks preceding the vote;
2) $\frac{p_{fav}^u}{s^u} > \frac{2}{3}$ a condition that holds iff the number of tweets posted by u in favor of his/her preferred faction is greater than twice the sum of the other tweets posted by u supporting different factions.

The $polarize$ function returns a score vector v_s^u containing the polarization degree of user u towards each considered faction in F; for this reason, $|v_s^u|$ is equal to $|F|$. In particular, we compared three different heuristic functions:

- H^*: it computes the contribution c_{fav}^u of the user u to his/her favorite faction p_{fav}^u, defined as $\frac{p_{fav}^u}{s^u}$ and returns a vector of real numbers containing c_{fav}^u in position fav^u, 0 otherwise.
- H_1: it returns a binary vector containing 1 in position fav^u, 0 otherwise.
- H_2: it returns a vector of real number containing for each faction $f \in F$ the contribution of the user u to f.

F. Results visualization

Results visualization is performed by the creation of infographics aimed at presenting the results in a way that is easy to understand to the general public, without providing complex statistical details that may be hard to understand to the intended audience. The graphic project is grounded on

Iteration	Tweet	Class
It. 0	American who loves her country #VoteTrump	Pro-Trump
	Guys, she's right! It's time!!! #GoVote #ImWitHer	Pro-Clinton
	Women detail sexual allegations against Trump #sex #woman #ImWitHer	Pro-Clinton
	List of Trump's accusers and their allegations #misconduct #sex #woman	*Unclassified*
	Hillary Clinton used personal email for government business #email #hillary #VoteTrump	Pro-Trump
	How Hillary Clinton used her personal email #email #hillary #scandal	*Unclassified*
	Hillary supports immigration reform with a pathway to citizenship #hillary #immigration	*Unclassified*
	A wall between the U.S. and Mexico #trump #mexico #immigration	*Unclassified*
	...	...
It. 1	List of Trump's accusers and their allegations #misconduct #sex #woman	Pro-Clinton
	How Hillary Clinton used her personal email #email #hillary #scandal	Pro-Trump
	Hillary supports immigration reform with a pathway to citizenship #hillary #immigration	*Unclassified*
	A wall between the U.S. and Mexico #trump #mexico #immigration	*Unclassified*
...	...	
It. 2	Hillary supports immigration reform with a pathway to citizenship #hillary #immigration	Pro-Clinton
	A wall between the U.S. and Mexico #trump #mexico #immigration	Pro-Trump
...	...	

TABLE II
EXAMPLE OF USE OF POST CLASSIFICATION ALGORITHM ON TWEETS OF 2016 U.S. PRESIDENTIAL ELECTION (ITERATIONS 0-2).

some of the most acknowledged and ever-working principles underpinning a 'good' info-graphic piece. In particular, we follow three main design guidelines: $i)$ preferring a visual representation of the quantitative information to the written one; $ii)$ minimizing the cognitive efforts necessary to decoding each system of signs; $iii)$ structuring the whole proposed elements into graphic hierarchies [22].

Displaying quantitative information by visual means instead of just using numeric symbols - or at least a combination of the two approaches - has been proven extremely useful in providing a kind of sensory evidence to the inherent abstraction of numbers, because this allows everybody to instantly grasp similarities and differences among values. In fact, basic visual metaphors (e.g., the largest is the greatest, the thickest is the highest) enable more natural ways of understanding and relating sets of quantities [23].

V. IMPLEMENTATIVE DETAILS OF THE CLASSIFICATION MODELS

The methodology proposed in this work has been implemented in Python using the framework *Keras*[1] with *TensorFlow*[2] as back-end for the creation of neural networks. Among the other libraries, we cite *sklearn*[3], used for the calculation of the confusion matrix and the resulting measures, and its library *imblearn*, used to deal with the class-imbalance problem.

The classification models we created are based on the *multi-layer perceptron*, a feed-forward neural network described in Section III-A. Several implementation choices have been done, as described in the following.

The input layer counts a number of neurons equal to the cardinality of the current dictionary used for the input vectorization step. According to the *universal approximation theorem* [24], only one fully connected hidden layer has been used, which enables to approximate functions defined on compact sets of $\mathbb{R}^n$. The choice of a single hidden layer also limits the possibility of overfitting and simplifies the learning process with respect to a deeper network. Following a commonly used rule-of-thumb, this layer presents 2/3 of the input neurons.

The activation function used within the hidden layer is *ReLU* (*Rectified Linear Unit*), which is defined as $R(z) = max(0, z)$. It has many interesting properties, such as: $i)$ biological plausibility, that is the sparse activation of artificial neurons mimics what happens in biological systems; $ii)$ scale invariance; $iii)$ faster training times compared to other activation functions (e.g.,$tanh$ or $sigmoid$); $iv)$ robustness against common issues (e.g., weight saturation, gradient vanishing). As regularization mechanism, a dropout layer has been inserted after the ReLU layer with a drop out rate fixed at 5%.

The output layer has a number of neurons equal to that of candidates/factions that have been considered. To distribute the probability that a post is assigned to the different factions, a normalized exponential probability distribution function, namely *softmax*, has been used.

The training phase has been carried out using: early stopping callbacks to prevent overfitting, accuracy as the main metric, the categorical cross-entropy loss function (i.e., an extension to the multiclass case of the most famous binary version), and the optimization algorithm ADAM [21]).

VI. A CASE STUDY

Here we discuss a case study carried out to analyze the polarization of a large number of Twitter users during the 2018 Italian general election. Twitter users have been classified using the polarization rules extracted from our methodology and the results have been compared with: $i)$ official results after the vote; $ii)$ main opinion polls collected before voting.

[1]https://keras.io/
[2]https://www.tensorflow.org/
[3]https://scikit-learn.org/

Iteration	Tweet input	Classified ($\mathbf{C^i}$)	Unclassified ($\mathbf{U^i}$)	Perc. of class. tweets	$\frac{\lvert\mathbf{C^i}\rvert}{\lvert\mathbf{U^{i-1}}\rvert}$	Accuracy
0	60,782	3,072	57,710	5.1%	5.1%	-
1	57,710	14,676	43,034	24.1%	25.4%	0.916
2	43,034	4,677	38,357	7.7%	10.9%	0.990
3	38,357	1,572	36,785	2.6%	4.1%	0.992
Total	60,782	23,997	36,785	39.5%	-	-

TABLE III

Partial results at iteration-level.

	LEGA%	PD%	M5S%	FI%	LogAcc	MAPE	MAE	$\mathbf{R^2}$
Real percentages	17.37	18.72	32.68	14.01	-	-	-	-
Averages of opinion polls	13.40	22.80	28.10	16.40	0.81	0.19	3.74	0.72
H*	***18.45***	***19.89***	***31.64***	***12.80***	***0.94***	***0.06***	***1.13***	***0.98***
H1	18.66	19.30	33.34	11.49	0.92	0.08	1.26	0.97
H2	19.37	21.93	29.34	12.14	0.87	0.13	2.61	0.86

TABLE IV

Obtained percentages and performance evaluation.

Italians voted to elect 630 deputies and 315 elected senators of the XVIII legislature: the results decreed the center-right coalition as the most voted, with about 37% of the preferences, while the most voted list, was the *Movimento 5 Stelle*, which received over 32% of votes. The electorate was 50,782,650 voters for the Chamber of Deputies and 46,663,202 for the Senate[4], with a turnout of about 73%, the lowest in Italian republican history.

In order to assess the validity of the proposed methodology, the analysis carried out focused on the four political factions that were most successful with public opinion, in decreasing order of consensus: *M5S* (*Movimento 5 Stelle*), *PD* (*Partito Democratico*), *LEGA*, *FI* (*Forza Italia*). In the following, we show how the classification model has been trained and the main results achieved.

A. Models training and iteration-level results

The algorithm described in Section IV-D has been used to classify 60,782 tweets posted by 21,833 users.

For improving data representativeness only tweets written in Italian have been considered. Moreover, the following positive faction keywords have been used:

- $K^{\oplus}_{M5S}$={#iovotom5s,#m5salgoverno,#dimaiopresidente}
- $K^{\oplus}_{PD}$={#sceglipd,#iovotopd,#pdvinci}
- $K^{\oplus}_{LEGA}$={#4marzovotolega,#iovotolega,#salvinipremier}
- $K^{\oplus}_{FI}$={#berlusconipresidente,#votoforzaitalia, #4marzovotoforzaitalia}

The threshold th and the minimum increment eps have been set to 0.9 and 5% respectively. In our test, the post classification algorithm terminated in 4 iterations by annotating 23,997 tweets, which represents about 39.5% of the total. Table III shows the obtained results at each iteration by specifying the number of classified and unclassified tweets, the total percentage of classified tweets, the ratio $\frac{|C^i|}{|U^{i-1}|}$ and the accuracy of the neural networks.

B. Polarization of users and final results

The algorithm described in Section IV-E has been used for analyzing the users who have written the 23,997 classified tweets so as to determine their polarization degree towards the considered factions. The algorithm has been executed using the *filter* function and the different *polarize* heuristics (H^*, H_1 and H_2) described in Section IV-E.

Table IV shows a comparison between the official results after the vote, the average of the latest opinion polls and the percentages obtained through the main heuristic H^* and the alternative ones, H_1 and H_2. We evaluated the accuracy through different statistical indexes, comparing the obtained results with the latest opinion polls published before the elections. Considering the four most supported parties and using the heuristic H^*, our methodology obtained the following approval percentages: LEGA 18.45%, PD 19.89%, M5S 31.64%, FI 12.80%. These results are extremely close to the real ones (i.e., LEGA 17.37%, PD 18.72%, M5S 32.68%, FI 14.01%), even more than the average of polls. In addition, the obtained results are characterized by very good values of log accuracy ratio and R^2, as well as a negligible value of mean absolute and percentage errors.

Figure 3 shows the comparison between the results achieved with the application of the developed methodology and the H^* heuristic, the real ones and the average of opinion polls carried out by YouTrend[5], a web magazine which focuses on social, economical and political trends. The results reported in the figure show clearly the closeness between the real percentages and the predicted ones, which are more accurate than the average of polls for every considered faction. In particular, our methodology achieved a mean average error (MAE) of

[4]http://www.interno.gov.it/it/notizie/elezioni-2018-come-vota-corpo-elettorale-tessera-elettorale

[5]`https://www.youtrend.it/`

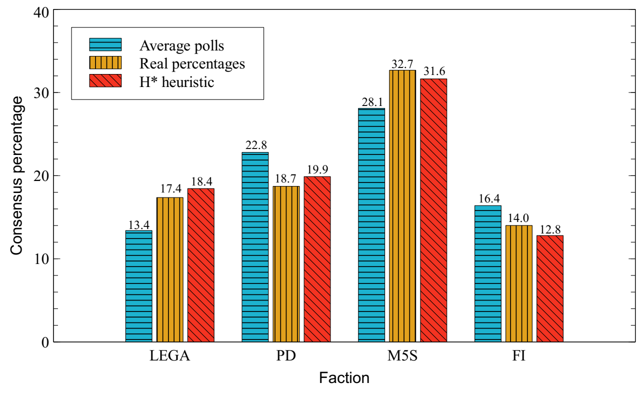

Fig. 3. Comparison between the obtained results, the real ones and the average of opinion polls.

1.13 percentage points and a determination coefficient R^2 very close to 1. Instead, opinion polls achieved a MAE of 3.74 percentage points and a R^2 of 0.72, which confirm the ability of the proposed methodology to forecast election results.

VII. Conclusion

The huge amount of data that comes from the interaction of people on the main social media, the so-called Big Social Data, has encouraged the development of a wide range of techniques and methodologies related to social media analysis. In this paper, we presented a new methodology based on feed-forward neural networks for estimating - in an almost automatic way - the polarization of public opinion during a political events. It allows to calculate the consensus of opposing parties among social users.

To validate the proposed methodology, it has been applied to a real case study, the Italian elections of 4 March 2018. In particular, Italian Twitter users have been classified using the polarization rules extracted from our methodology and the results have been compared with opinion polls collected before voting and with the actual results obtained after the vote. The achieved results are very close to the real ones and are significantly more accurate than the average of the opinion polls, revealing the high accuracy and effectiveness of the proposed approach.

Acknowledgment

This work has been partially supported by the SMART Project, CUP J28C17000150006, funded by Regione Calabria (POR FESR-FSE 2014-2020) and by the ASPIDE Project funded by the European Union's Horizon 2020 Research and Innovation Programme under grant agreement No 801091.

References

[1] L. Belcastro, F. Marozzo, D. Talia, and P. Trunfio, "Big data analysis on clouds," in *Handbook of Big Data Technologies*, A. Zomaya and S. Sakr, Eds. Springer, December 2017, pp. 101–142.

[2] E. Cesario and et al., "Analyzing social media data to discover mobility patterns at expo 2015: Methodology and results," in *The 2016 International Conference on High Performance Computing and Simulation (HPCS 2016)*, Innsbruck, Austria, 18-22 July 2016.

[3] E. Olshannikova, T. Olsson, J. Huhtamäki, and H. Kärkkäinen, "Conceptualizing big social data," *Journal of Big Data*, vol. 4, no. 1, p. 3, 2017.

[4] D. Talia, P. Trunfio, and F. Marozzo, *Data Analysis in the Cloud*. Elsevier, October 2015, iSBN 978-0-12-802881-0.

[5] I. El Alaoui, Y. Gahi, R. Messoussi, Y. Chaabi, A. Todoskoff, and A. Kobi, "A novel adaptable approach for sentiment analysis on big social data," *Journal of Big Data*, vol. 5, no. 1, p. 12, 2018.

[6] J. Ramteke, S. Shah, D. Godhia, and A. Shaikh, "Election result prediction using twitter sentiment analysis," in *2016 international conference on inventive computation technologies*, vol. 1, 2016, pp. 1–5.

[7] R. Jose and V. S. Chooralil, "Prediction of election result by enhanced sentiment analysis on twitter data using classifier ensemble approach," in *2016 international conference on data mining and advanced computing (SAPIENCE)*, 2016, pp. 64–67.

[8] T. Mahmood, T. Iqbal, F. Amin, W. Lohanna, and A. Mustafa, "Mining twitter big data to predict 2013 pakistan election winner," in *INMIC*, 2013, pp. 49–54.

[9] M. D. Conover, J. Ratkiewicz, M. Francisco, B. Gonçalves, F. Menczer, and A. Flammini, "Political polarization on twitter," in *Fifth international AAAI conference on weblogs and social media*, 2011.

[10] J. DiGrazia, K. McKelvey, J. Bollen, and F. Rojas, "More tweets, more votes: Social media as a quantitative indicator of political behavior," *PloS one*, vol. 8, no. 11, p. e79449, 2013.

[11] K. Jahanbakhsh and Y. Moon, "The predictive power of social media: On the predictability of us presidential elections using twitter," *arXiv preprint arXiv:1407.0622*, 2014.

[12] F. M. F. Wong, C. W. Tan, S. Sen, and M. Chiang, "Quantifying political leaning from tweets, retweets, and retweeters," *IEEE transactions on knowledge and data engineering*, vol. 28, no. 8, pp. 2158–2172, 2016.

[13] D. Chin, A. Zappone, and J. Zhao, "Analyzing twitter sentiment of the 2016 presidential candidates," *American Journal Of Science and Research*, 2016.

[14] T. Graham, D. Jackson, and M. Broersma, "New platform, old habits? candidates' use of twitter during the 2010 british and dutch general election campaigns," *New media & society*, vol. 18, no. 5, pp. 765–783, 2016.

[15] F. Marozzo and A. Bessi, "Analyzing polarization of social media users and news sites during political campaigns," *Social Network Analysis and Mining*, vol. 8, no. 1, p. 1, 2018.

[16] L. Oikonomou and C. Tjortjis, "A method for predicting the winner of the usa presidential elections using data extracted from twitter," in *2018 South-Eastern European Design Automation, Computer Engineering, Computer Networks and Society Media Conference (SEEDA_CECNSM)*. IEEE, 2018, pp. 1–8.

[17] K. Jaidka, S. Ahmed, M. Skoric, and M. Hilbert, "Predicting elections from social media: a three-country, three-method comparative study," *Asian Journal of Communication*, pp. 1–21, 2018.

[18] M. K. Olorunnimbe and H. L. Viktor, "Tweets as a vote: Exploring political sentiments on twitter for opinion mining," in *International Symposium on Methodologies for Intelligent Systems*. Springer, 2015, pp. 180–185.

[19] A. K. Jain, J. Mao, and K. Mohiuddin, "Artificial neural networks: A tutorial," *Computer*, no. 3, pp. 31–44, 1996.

[20] I. A. Basheer and M. Hajmeer, "Artificial neural networks: fundamentals, computing, design, and application," *Journal of microbiological methods*, vol. 43, no. 1, pp. 3–31, 2000.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[22] E. Cesario, C. Congedo, F. Marozzo, G. Riotta, A. Spada, D. Talia, P. Trunfio, and C. Turri, "Following soccer fans from geotagged tweets at fifa world cup 2014," in *Proc. of the 2nd IEEE Conference on Spatial Data Mining and Geographical Knowledge Services*, Fuzhou, China, July 2015, pp. 33–38, iSBN 978-1-4799-7748-2.

[23] E. R. Tufte, *The Visual Display of Quantitative Information*. Cheshire, CT, USA: Graphics Press, 1986.

[24] B. C. Csáji, "Approximation with artificial neural networks," *Faculty of Sciences, Etvs Lornd University, Hungary*, vol. 24, p. 48, 2001.