Software actors for continuous social media analysis

Paolo Fornacciari, Monica Mordonini, Agostino Poggi, Michele Tomaiuolo
Università di Parma, Italy,
{paolo.fornacciari,monica.mordonini,agostino.poggi,michele.tomaiuolo}@unipr.it

***Abstract*—Social media analysis is rapidly becoming a widespread tool for various applications. Motivations for interest range from commercial marketing to the monitoring of social trends and political opinions. This work presents an actor-based platform, augmented with machine learning algorithms. To evaluate its possible application for the continuous analysis of social media streams, we also present a case study about sport. We argue that the actor paradigm allows modeling well this kind of scenario, and it can scale to more complex applications, where multiple actors can cooperate to realize structured analysis tools. Results also show the advantage of a system for continuous analysis, which can learn from new data and improve its internal classification model.**

***Keywords*-Data analysis; actor-based systems; machine learning; social media.**

I. Introduction

Currently, people use online social networks for more and more types of interactions, including recreational interests, family relationships and work activities [1]. Consequently, there is a strong interest to monitor such online platforms for obtaining some early insight into societal changes, in both online and offline life. Since the use of social networks is much differentiated, also the kinds of analysis and studies that are being conducted vary. While companies are mainly interested in the public fidelity to brands, products and characters, politicians and decision makers require instead the analysis of emerging social trends and political opinions.

This paper presents a system for continuous monitoring and analysis of social streams. The system augments an actor based software framework with a non-relational data repository and machine learning algorithms. The actor model provides suitable features for both simplifying the development of large and distributed complex systems and guaranteeing scalable and efficient applications.

There are a multitude of actor oriented libraries and languages, and each of them implements some variants of actor semantics. ActoDeS (Actor Development System) is a framework which allows developers to build heterogeneous systems [2]. It allows merging thread-based programming, which eases the development of programs, with event-based programming, which is far more practical to develop large and efficient concurrent systems, but also is more difficult to use.

The ActoDeS framework can be adapted for the use on social media. In particular, in this work we have realized a system for monitoring some social streams. The use case we have chosen is the topic "sport", which we have observed on Twitter, for gaining information about the varying popularity of some specific sports, during the analyzed period.

An automatic classifier, based on configurable supervised learning algorithms, is trained on some initial dataset. Then, it is applied on the stream of data collected at runtime, to classify new instances. Downloaded data is annotated and stored in some collections in a non-relational database, as JSON objects. Additionally, new data can be used to augment the initial training set. This way, the classifier can use its own classification to produce new data to learn from and improve its own internal prediction model.

Before discussing in detail the implemented system, in the next section we present some related work. Section 3 then introduces the underlying software framework and provides details about the system architecture and its implementation. Sections 4 describes the case study and the methodology that we have used to collect and analyze data. Section 5 presents the experimental results obtained on the case study. Section 6 discusses how the system is being used and can be further extended, to realize a more complex classification system. Finally, section 7 concludes the paper by discussing the current features of the system and evaluating the main development lines.

II. Related Work

The focus of social network analysis (SNA) is on relationships among the members of social communities. This analysis is fundamental in the social and political sciences, as well as in economics and industrial engineering. Historically SNA is a research strategy that allows researchers to quantify the pattern of relations among a set of actors. A comprehensive introduction to the theory and practice of network analysis in the social sciences is provided in [3]. In addition to the structural analysis of a social entity, it is interesting to study the emotional components of community members. This is a research field that takes the name of Opinion Mining and Sentiment Analysis. The growing availability and popularity of online social media and personal blogs gives new opportunities and challenges, in the use of information technologies and machine learning to provide tools to understand the opinions of others. A survey of these techniques is shown in [4]. Examples of applications of these methods are shown in [5], where a corpus of geotagged tweets is used for estimating happiness in Italian cities, or in [6] where the US mood throughout the day are inferred from an analysis performed on Twitter data. In [7] authors present a possible combined approach between Social Network Analysis and Sentiment Analysis. In particular, a sentiment is associated to the nodes of the graphs showing the social connections, and this may highlight the potential correlations. The growth and pervasiveness of online communities allow different and new possibilities to interact with them. For example, social networking systems blur the distinction between the private and working spheres, and users are known to use such systems both at home and on the work place, both professionally and with recreational goals [8]. Moreover, social networking has a large potential for easing collaboration, also across organizational boundaries, but effective e-collaboration through social networks requires the development of open and interoperable systems. This could create some difficulties in the domains of privacy or confidentiality [9]. Some work has been done in the analysis of undesirable behavior such as trolling, such as in [10]. Like other organizations, sport industry groups including athletes, teams, and leagues use social media to share information about and promote their products. In [11], authors explore how sporting event organizers and influential Twitter users spread information through the online social network. In general, data coming from online social media is an example of Big Data, which often requires the use of distributed and scalable software tools. Several actor oriented libraries have been proposed in recent years, which could be used to implement a distributed sentiment analysis system (see, for example Scala [12] and Akka[13]). Scala is an object-oriented and functional programming language that provides an implementation of the actor model unifying thread based and event based programming models. In fact, in Scala an actor can suspend with a full thread stack (receive) or can suspend with just a continuation closure (react). Therefore, scalability can be obtained by sacrificing program simplicity. Akka is an alternative toolkit and runtime system for developing event-based actors in Scala, but also providing APIs for developing actor-based systems in Java. One of its distinguishing features is the hierarchical organization of actors, so that a parent actor that creates some children actors is responsible for handling their failures.

III. System architecture

ActoDES is a software framework which aims at simplifying the development of complex distributed systems, implementing an actor model [2]. Actors are autonomous and concurrent objects, each characterized by a state and a behavior and the ability to interact with other agents through the exchange of asynchronous messages. After the analysis of its incoming messages, an actor can send more messages to itself or to others, create new actors, update its state, change its behaviors, terminate its own execution, etc. Each behavior can define a policy for handling incoming messages, through handlers called "cases". Each case can only process messages corresponding to a specific pattern. An actor space is intended as a container offering the services needed for the correct execution of a set of actors. In particular, it includes two types of actors: a scheduler and a service provider. The former has the duty to handle the concurrent execution of actors, while the latter provides runtime services, needed by actors to complete their tasks.

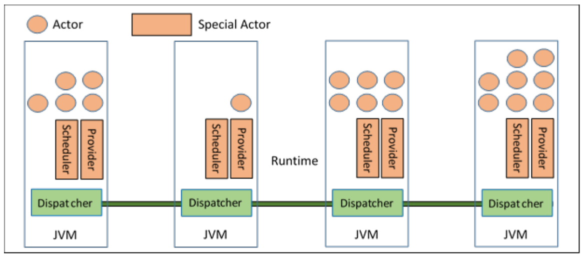

Fig. 1. Distributed ActoDeS application architecture.

In previous works, we have discussed the role of agents and actors for the analysis and simulation of social networks [14][15] and for building social networking platforms [16][17]. In particular, these platforms may provide more control to users for managing and protecting their own data [9][18]. This project is centered on the realization of new services, providing additional functionalities to actors, for the continuous analysis of social streams. One of such services is a Mailer service, to which actors can send subscription requests, for a set of addresses. A mailer service is useful in developing collaborative applications with actors as shown in [19]. This service has the task to receive incoming messages into a specific mailbox, and forward them to subscriber actors. The actors can eventually handle the messages differently, according to their own behaviors. Another module introduced in the system is a Twitter service, to interact with the widespread social platform. The Twitter service allows an actor to send various kinds of requests:

- User timeline, to obtain the recent tweets of a specified user and save them in a local storage system
- Content query, to obtain recent tweets published on Twitter and selected according to some constraints specified by the actor
- Stream, to receive messages published on the platform during the execution; tweets are obtained in the form of JSON objects, which are then store in NoSQL repository (namely a MongoDB database [20])

Leveraging ActoDES and the additional mentioned modules, we have built a software system that can be used to track and study a certain topic, with an architecture that can be extended to different cases and to problems that are more complex. The implemented system allows to:

- Download tweets published by a given user
- Search data published by the community, with the ability to set all relevant parameters
- Create collections in MongoDB, to store obtained JSON objects
- Join different collections to create a training set, in "arff" format
- Training a classifier on the produced training set
- Save and load a classifier model from a file
- Start observing a stream of tweets, matching a given query
- Get the results of a classifier, on the new tweets

Additionally, the system is able to filter out repeated messages, or very similar messages. In fact, these messages can alter the classifier model, if they are introduced repeatedly into the training set. For this aim, the Levenshtein distance has been used, to measure the similarity of texts [21]. If some tweets are measured above a configurable threshold, then they are blocked from further processing.

The classifier is an entity that requires a training set of initial instances to create an internal prediction model, and then it can learn to classify also new instances. To evaluate the performance of a classifier, an annotated test set can be used. To deal with text, Naive Bayes Classifiers usually provide good results [22] [23] [24]. Their simplified model is based on the hypothesis of independence of features, i.e., the presence of a feature is not correlated to the presence of other features.

Being the system dedicated to the continuous analysis of social media streams, it allows feeding the classifier with new data, for augmenting its training set at runtime. For evaluation purposes, it is also possible to use two different classifiers, used to classify the same data in a parallel architecture. The first classifier can be trained once, in a static way, with the initial training set. The second one can learn also from new data, by updating its own training set with new instances. In particular, it is possible to set a confidence threshold, and add new instances, which are classified with enough confidence, incrementally to the original training set. In this sense, the second classifier is continuously self-training on the data stream it receives.

Additionally, to monitor the classification results, a web module has been introduced. The web interface updates its representation dynamically, showing basic results about collected tweets or observed streams. In particular, the interface shows:

- A pie chart, to show the total number of tweets that have been eventually classified for each class, since the beginning of the process
- A line chart, to show how many tweets are being classified for each class during each timeframe of the observed period
- The results of classifiers for all tweets
- The frequency of classifications of latest tweets
- Total number of tweets
- Number of repeated tweets (retweets)
- Number of tweets discarded, for the process of online training (described later)
- Localization of tweets, for which some form of geographic information is available

An additional level of analysis can be based on the geolocalization of tweets and users. In the graphical interface, a map allows to position tweets that are annotated, in different ways, with geographical coordinates. For each localized tweet, a marker is added on the map, representing the class associated with the tweet after the classification process. During the analysis of data, we noted that only a small minority of tweets are annotated with precise geographical coordinates, i.e., only from mobile users who enable the localization feature of their Twitter app. Fortunately, geographical data can be obtained in different ways. In particular, two additional fields, relevant for positioning a tweet, are its “place” and “location”. Place can contain information about a user’s place of residence, according to his/her own choice. This information is requested

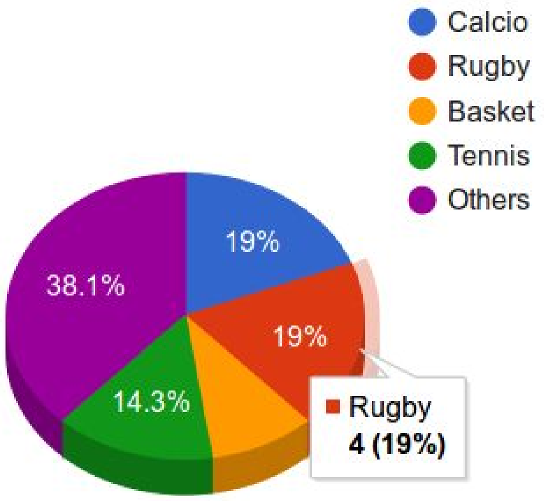

Fig. 2. Example of pie chart, for classified tweets.

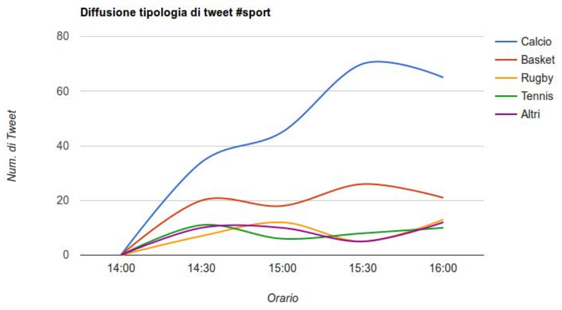

Fig. 3. Example of line chart, for classified tweets.

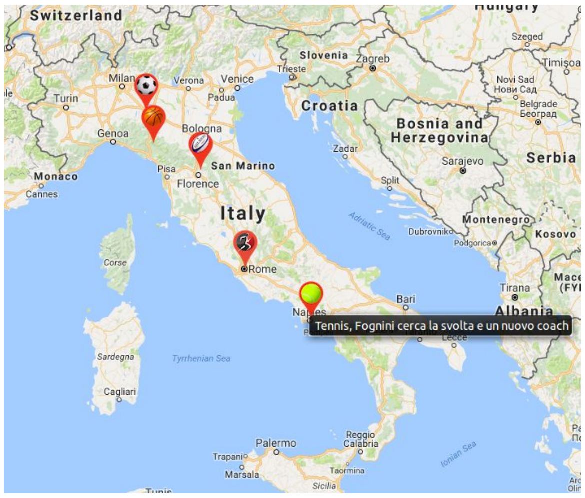

Fig. 4. Example of localized tweets.

during the registration and tends to be static, apart from manual a-posteriori modifications. Instead, location is requested before the publication of a tweet, optionally and with a precision level that can be selected by the user. Both the place and location fields contain textual information for cities and regions, which require the use of a “reverse geocoding” service to obtain an approximation for geographical coordinates.

IV. A CASE STUDY

We have used “sport” as a topic for conducting a case study. In fact, sport is a widespread topic for which it is easy to find abundance of data and news, in virtually any period. This study, in particular, is focused on the classification of tweets among few chosen sports: soccer, rugby, basket, tennis. For each class, a proper training set is needed by the system, in the form of an “arff” file, with texts annotated according to the selected classes. An additional class is introduced for tweets marked as “generic/others”, which deal with sport but not specifically about the four classes of this study. For creating the training set, we have collected tweets published on Twitter public channels, i.e., tweets with a specific hashtag, in this case the name of the specific sport of interest. This way, however, a quantity of collected data is not really related to the supposed topic, and can thus worsen the classification results. Consequently, we have manually filtered the obtained data, for removing spurious tweets and thus creating a better classification model. Moreover, classical prefiltering techniques are useful to obtain higher precision. Finally, all selected data has to be joined into a whole “arff” file, containing instances of the five classes. In this case, the training set is composed of around 230-250 instances for each class, summing up to 1140 instances in total. The number of instances in the training set is not large. In fact, we have decided to create a small functioning training set, and then possibly augment it automatically at runtime. Small initial training set allows appreciating better any improvement in the quality of the classification model, as it eventually matures.

For downloading tweets, Twitter APIs [25] expose a public stream interface. It allows to be notified at runtime about new tweets, as soon as they are published. These messages can be filtered according to their textual content, language and many other parameters. Thus, an actor can use these filters to receive and process only the tweets of its own interest. The present study is based on Italian tweets, with the generic hashtag #sport.

More in detail, a listener is used to handle new tweets. At the reception of each message, the listener performs the following actions:

- Add the tweet, as a JSON object, to a collection in the NoSQL data repository
- Extract or estimate the geographical position of the tweet
- Update files containing the classification results
- Update the training set, augmenting it with new instances
- Update aggregated data on the stream of tweets under analysis

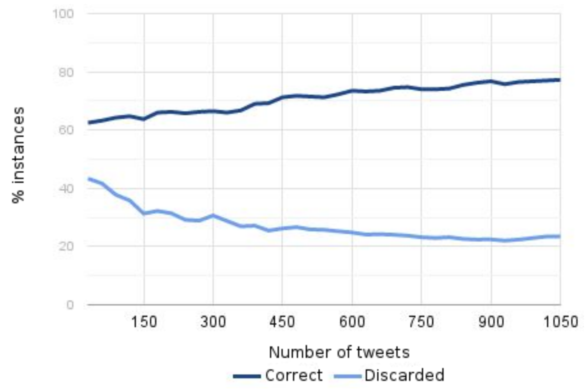

Fig. 5. Improvement of classification quality.

V. RESULTS

To evaluate results, in the case study about sport, two classifiers are executed in parallel. Both classifiers are initially trained on the same data set, and both have to predict the class of the same tweets, published on the #sport channel (hashtag). Only one of these two classifiers uses its own classification results to augment its own training set, continuously. A tweet is appended to the training set at the condition that it is classified with a sufficient confidence level. In this study, the threshold is set at 68%. Once beginning to listen on the stream, the classification quality is re-evaluated automatically after the reception of each 30 tweets. Both classifiers are tested against the same test set, composed of 400 instances, distributed almost equally among the various classes. This way, it is possible to track any change in the quality of the online learning classifier. In fact, after collecting 1050 tweets and using a selection of them to update the training set, the level of correct classifications has raised from 62.5% to 76.0%, eventually. Moreover, the quantity of discarded tweets, i.e., for which a class cannot be defined with sufficient confidence, decreases during time. This is another evidence that the use of new data can improve the classification model. It has to be noted that received tweets, with #sport hashtag, often contains very confused information, thus this level of quality improvement over time could not be easily predicted.

About the geo-localization of tweets, we have noted that tweets associated directly with geographical coordinates are quite few, around 5% in our study. However, using the place and location fields and a reverse geocoding service (Google Maps API, namely), around 50% of all tweets can be associated with an estimation of their geographical position.

VI. FURTHER WORK

The basic case described in this work has been useful to experiment with actor technology for social stream analysis and online learning. However, the sport scenario does not exploit the full benefits of a powerful actor based system. In fact, we are already leveraging this experience for building more complex systems. The very next step we are taking is to implement a runtime adaptive system for automatic emotion

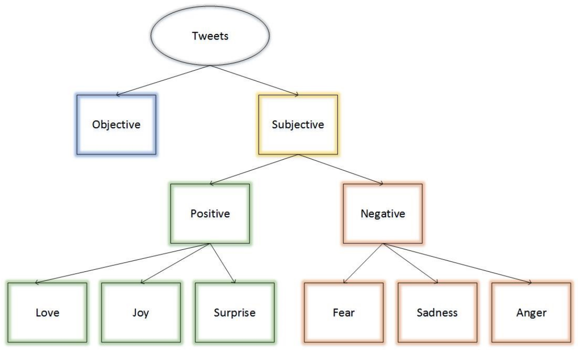

Fig. 6. Improvement of classification quality.

classification. Starting from the basic concept of sentiment analysis, we have already demonstrated that it is possible to discriminate more precise emotions associated with a post, rather than simple positive or negative sentiments [26]. In particular, the basic model of sentiment analysis can be extended in the direction of specifying the emotions that characterize subjective tweets. Our previous works have applied Parrott's socio-psychological model to social media classification [27]. According to this model, all human feelings are divided into six major states (three positive and three negative):

- Positive feelings of love, joy, and surprise
- Negative feelings of fear, sadness, and anger

Thus, into the ActoDeS framework, we are building a hierarchical classifier, composed of multiple coordinated actors, representing the different levels and branches of the whole classifier. The hierarchy of classification is shown in Figure 6. In general, hierarchical classification is based on the consistent application of multiple classifiers, organized in a tree-like structure. In the particular case of Parrot's emotions, a first step uses a binary classifier that determinates the subjectivity/objectivity of a tweet. The second step further processes all instances that have been identified as subjective. It uses another binary classifier that determinates the polarity (positivity/negativity) of a tweet. Depending on the polarity assessed at the previous level, the third step classifies the specific emotion expressed in the text (love, joy or surprise for positive tweets; fear, sadness or anger for negative tweets). All these classifiers are embodied by actors, which can operate on a stream of tweets, analyzing them and, finally, learning from them.

The use of a hierarchical structure of classification is motivated by previous results, which show that the domain knowledge embedded into the hierarchical classifier makes it more accurate than the flat classifier. Moreover, the automatic construction of a training set with data acquired from social media has demonstrated to be a viable solution. In particular, in the case of sentiment analysis and emotion classification, it is possible to filter the training set automatically. This eventually improves the quality of the final classifier, with respect to a "blind" collection of raw data based only on the hashtags. In fact, the results obtained with this approach [26] are comparable with those found in similar works [28].

VII. Conclusions

In this work, we have proposed a system for the continuous analysis of social streams. The system is based on ActoDeS, a software framework that allows the development of efficient large actor based systems. It combines the possibility to use different implementations of the components driving the execution of actors with the delegation of the management of the reception of messages to the execution environment.

For the purpose of social media analysis, we have integrated ActoDeS with a non-relational data storage system and machine learning algorithms. We have tested the system with satisfaction for automatically classifying tweets about sport. We confronted the performances of two classifiers: one with static training and one with online training, i.e., augmenting its training data with new tweets obtained and analyzed at runtime. Though data obtained from social streams is often noisy and confused, nevertheless the second classifier quickly gained a significant advantage over the first one.

We are extending this kind of system for more complex use cases, requiring the use of structured classifiers. In particular, we are interested in applying a hierarchical system for emotion detection to a continuous stream of data. For all these kinds of applications, in our experience, the actor model provide quite handy features which simplify the system development and its usage for practical monitoring and analysis, in particular when the stream requires good scalability.

References

[1] E. Franchi, A. Poggi, and M. Tomaiuolo, "Social media for online collaboration in firms and organizations," *International Journal of Information System Modeling and Design (IJISMD)*, vol. 7, no. 1, pp. 18–31, 2016.

[2] F. Bergenti, A. Poggi, and M. Tomaiuolo, "An actor based software framework for scalable applications," *Lecture Notes in Computer Science (LNCS)*, vol. 8729, pp. 26–35, 2015.

[3] J. Scott, *Social network analysis*. Sage, 2012.

[4] B. Liu, "Sentiment analysis and opinion mining," *Synthesis lectures on human language technologies*, vol. 5, no. 1, pp. 1–167, 2012.

[5] L. Allisio, V. Mussa, C. Bosco, V. Patti, and G. Ruffo, "Felicità: Visualizing and estimating happiness in italian cities from geotagged tweets." *ESSEM@ AI* IA*, vol. 1096, pp. 95–106, 2013.

[6] A. Mislove, S. Lehmann, Y.-Y. Ahn, J.-P. Onnela, and J. N. Rosenquist, "Pulse of the nation: Us mood throughout the day inferred from twitter," *Northeastern University*, 2010.

[7] P. Fornacciari, M. Mordonini, and M. Tomaiuolo, "Social network and sentiment analysis on twitter: Towards a combined approach," in *KDWeb 2015, 1st International Workshop on Knowledge Discovery on the WEB*, ser. CEUR Workshop Proceedings, vol. 1489. CEUR-WS.org, 2015. [Online]. Available: http://ceur-ws.org/Vol-1489/paper-06.pdf

[8] G. Angiani, P. Fornacciari, E. Iotti, M. Mordonini, and M. Tomaiuolo, "Models of participation in social networks," in *Social Media Performance Evaluation and Success Measurements*. IGI GLobal, 2017, pp. 196–224.

[9] E. Franchi, A. Poggi, and M. Tomaiuolo, "Blogracy: A peer-to-peer social network," *International Journal of Distributed Systems and Technologies (IJDST)*, vol. 7, no. 2, pp. 37–56, 2016.

[10] J. Cheng, C. Danescu-Niculescu-Mizil, and J. Leskovec, "Antisocial behavior in online discussion communities," *arXiv preprint arXiv:1504.00680*, 2015.

[11] M. E. Hambrick, "Six degrees of information: Using social network analysis to explore the spread of information within sport social networks," *International Journal of Sport Communication*, vol. 5, no. 1, pp. 16–34, 2012.

[12] P. Haller and F. Sommers, *Actors in Scala*. Artima Incorporation, 2012.

[13] "Akka toolkit," http://akka.io/.

[14] F. Bergenti, E. Franchi, and A. Poggi, "Selected models for agent-based simulation of social networks," in *3rd Symposium on Social Networks and Multiagent Systems (SNAMAS 2011)*, 2011, pp. 27–32.

[15] F. Bergenti, E. Franchi, and A. Poggi, "Agent-based interpretations of classic network models," *Computational and Mathematical Organization Theory*, vol. 19, no. 2, pp. 105–127, 2013.

[16] A. Poggi and M. Tomaiuolo, "A dht-based multi-agent system for semantic information sharing," *Studies in Computational Intelligence*, vol. 439, pp. 197–213, 2013.

[17] A. Poggi and M. Tomaiuolo, "Integrating peer-to-peer and multi-agent technologies for the realization of content sharing applications," *Studies in Computational Intelligence*, vol. 324, pp. 93–107, 2011.

[18] E. Franchi, A. Poggi, and M. Tomaiuolo, "Information and password attacks on social networks: An argument for cryptography," *Journal of Information Technology Research (JITR)*, vol. 8, no. 1, pp. 25–42, 2015.

[19] S. Gallardo-Vera and E. Nava-Lara, "Developing collaborative applications with actors," in *Proceedings of the World Congress on Engineering and Computer Science*, vol. 1, 2015.

[20] K. Chodorow, *MongoDB: the definitive guide*. " O'Reilly Media, Inc.", 2013.

[21] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet physics doklady*, vol. 10, no. 8, 1966, pp. 707–710.

[22] K. P. Murphy, "Naive bayes classifiers," *University of British Columbia*, 2006.

[23] J. Chen, H. Huang, S. Tian, and Y. Qu, "Feature selection for text classification with naive bayes," *Expert Systems with Applications*, vol. 36, no. 3, pp. 5432–5435, 2009.

[24] P. Fornacciari, M. Mordonini, and M. Tomaiuolo, "A case-study for sentiment analysis on twitter," in *WOA 2015, From Objects to Agents*, ser. CEUR Workshop Proceedings, vol. 1382. CEUR-WS.org, 2015, pp. 53–58. [Online]. Available: http://ceur-ws.org/Vol-1382/paper8.pdf

[25] "Twitter api," https://dev.twitter.com/overview/api.

[26] G. Angiani, S. Cagnoni, N. Chuzhikova, P. Fornacciari, M. Mordonini, and M. Tomaiuolo, "Flat and hierarchical classifiers for detecting emotion in tweets," *Lecture Notes in Computer Science (LNCS)*, vol. 10037, pp. 51–64, 2016.

[27] W. G. Parrott, *Emotions in social psychology: Essential readings*. Psychology Press, 2001.

[28] D. Ghazi, D. Inkpen, and S. Szpakowicz, "Hierarchical versus flat classification of emotions in text," in *Proceedings of the NAACL HLT 2010 workshop on computational approaches to analysis and generation of emotion in text*. Association for Computational Linguistics, 2010, pp. 140–146.