Mining Long-term Topics from a Real-time Feed

Marijn ten Thij

Vrije Universiteit Amsterdam,
Faculty of Science,
Amsterdam, The Netherlands
Email: m.c.ten.thij@vu.nl

***Abstract*—In our current society, the availability of data has gone from scarce to abundant: huge volumes of data are generated every second. A significant part of these data are generated on social media platforms, which provide a very volatile flow of information. Leveraging the information that is buried in this fast stream of messages, poses a serious challenge. In this paper, we aim to distinguish all topics that are discussed in real-time in a social media feed by employing clustering and algorithmic techniques. We evaluate our approach by comparing the results to a post-hoc clustering approach.**

Keywords–Topic Detection and Tracking; Twitter; Cluster Analysis; Content analysis; First Story Detection.

I. INTRODUCTION

In recent years, social media have revolutionized the way people communicate and interact with each other. This development has transformed the Internet into a more personal and participatory medium, where social networking is the top online activity. The massive amount of data, that is accumulated as a result of these online interactions, discussions, social signals, and other engagements, forms a valuable source of information. In our current work, we focus on the application and leveraging of this information for a particular sector: the horticulture industry.

The horticulture industry is a traditional sector in which growers are focused on production, and in which many traders use their own transactions as the main source of information. Growers and traders, therefore, lack data about consumer trends and how the products are used and appreciated. This results in reactive management with very little anticipation to events in the future. Social media can provide the opportunities to enhance the market orientation of the horticulture industry. For example, tracking how and when the products of the industry are mentioned in a social media feed is an important addition to current techniques used in the horticulture industry to actively listen to customers. The feedback that is thus collected, can be used to understand, react, and provide value to customers.

Since the information from a social media feed is very volatile, it is important that the information is processed in real-time. To cope with this challenge of processing in real-time, we propose an algorithm to find and distinguish the aforementioned mentions in a real-time information feed. To do so, we define a story as the repeated and related mentions of a product in the real-time feed. Furthermore, we use the term topic for the content of these mentions within a story. In this paper, we base our algorithm on data that is scraped from Twitter. However, all parts of the algorithm can be easily modified to fit data scraped from other platforms, e.g., Instagram and Facebook, allowing for wider use of the designed approach. Using our algorithm, we are able to give an overview of what is being discussed in real-time with respect to the horticulture sector. This enables businesses to keep up with their reputation and customer satisfaction.

The lay-out of the paper is as follows. First, we discuss the related research in Section II. Then, we describe the dataset used for testing this filtering approach in Section III. Next, we employ clustering techniques to define a ground truth to test our filtering approach in Section IV, followed by the description of our filtering approach in Section V. The results of the comparison of these two approaches are then discussed in Section VI. Finally, we conclude the paper in Section VII with some discussion and opportunities for future work.

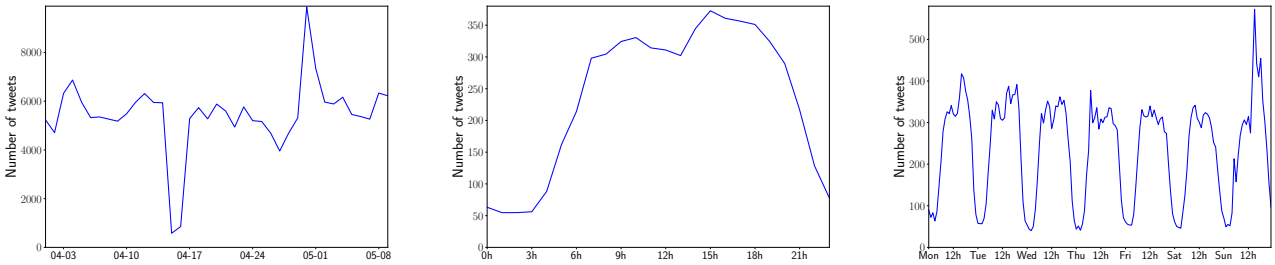
II. RELATED RESEARCH

The detection of emerging topics in a real-time information stream has been extensively studied. A good example is the Topic Detection and Tracking (TDT) research project [1], in which news items are combined in stories, that are tracked through time. Examples of TDT systems are the Europe Media Monitor [2], a platform that links news articles mentioning similar topics over time and across languages; and RTreporter [3] and Hotstream [4], two systems for breaking news detection and tracking in Twitter.

Based on the TDT project, Allan [5] defines five tasks that are part of topic detection and tracking, namely,

- Story Segmentation; dividing the transcript of a news show into individual stories.
- First Story Detection; recognizing the onset of a new topic in the stream of news stories.
- Cluster Detection; grouping all stories as they arrive, based on the topics they discuss.
- Tracking; monitoring the stream of news stories to find additional stories on a topic that was identified using several sample stories.
- Story Link Detection; deciding whether two randomly selected stories discuss the same news topic.

 ISBN: 978-1-61208-603-3



(a) Daily tweet numbers (b) Daily tweet rhythm (c) Weekly tweet rhythm

Figure 1. Overall statistics of the tweets used in this study. Figure 1a shows the daily number of tweets received by our scraper that were tagged as mentioning at least one product. Figures 1b and 1c show the average daily and weekly rhythms.

In our work, we focus on three of the five tasks, namely First Story Detection, Cluster Detection, and Tracking, to a real-time feed of Twitter messages. To perform the aforementioned tasks, which have some overlap, several approaches have been studied in the past. For instance, Weng and Lee [6] used clustering of wavelet-based signals for event detection, and Huang et al. [7] use a concept graph to discover topics by clustering the graph. In this paper, we use a clustering algorithm to find and track topics over time.

Several clustering algorithms have been developed over time, e.g., Affinity Propagation [8], Parameter-free Affinity Propagation [9], Spectral Clustering [10], DBSCAN [11], and Latent Dirichlet Allocation (LDA) [12]. A large body of studies have been devoted to adapting and extending LDA. For instance, Holz and Teresniak [13] employ the term co-occurence to track topics and topic change over time in news documents. Furthermore, Wang and McCallum [14] extend LDA to 'Topics Over Time' to incorporate time on top of term co-occurences. Staying in a similar scope as LDA, Swan and Allan [15] present a technique of topic detection on a corpus of documents based on co-occurences of Natural Language Processing (NLP) features extracted from the documents. In our study, we combine NLP-features of the messages in the information feed with the Affinity Propagation clustering algorithm, since it does not require the number of clusters as input parameter.

III. DATASET

The goal of our work is to develop a system that performs a real-time analysis of messages posted on Twitter, which we implement in *python*. Hence, we set up our own Twitter scraper. We scrape the tweets using the filter stream of the Twitter Application Programming Interface (API) [16]. Since we do not have access to the Twitter Firehose, we do not receive all tweets that we request due to rate limitations by Twitter [17]. Within these restrictions, we set up a stream with the goal to scrape as many Dutch tweets as possible. We use the filter stream with the options **language**, which we set to Dutch, and **track**, where a list of words must be defined. All tweets containing one of these words are caught by the Twitter API. Based on the number of occurences of these words in the dataset described in [18], we define a list of 400 general Dutch words (e.g., *'een, het, ik, niet, maar, die, de, bij, ook'*).

For this study, we do not use all tweets that we scraped in the way mentioned above. Since we are only interested in tweets that could be of value for the horticulture industry, we select a subset of these tweets that cover topics of interest to this industry, using a list of product names provided by our partners from GroentenFruit Huis[1] and Floricode[2]. The terms are split up into two lists: one containing fruits and vegetables, e.g., apple, orange, and mango, and the other containing flowers and plants, e.g., tulip, rose, and lily. We use the tweets that have mentioned at least one of the products on the lists that we obtained from April 1st 2017 12 AM through May 10th 2017 12 AM in Coordinated Universal Time (UTC). During this interval, we have not obtained any tweets from 7 AM on April 15th through April 16th at 6 PM, which is due to the down-time of our scraper. This down-time directly explains the decrease in the number of tweets that can be seen in Figure 1a, which shows the daily number of tweets that are tagged to mention at least one of the products of interest. Since we only consider Dutch tweets, we see a clear circadian rhythm in the number of interesting tweets per hour, both on the daily and weekly scale (shown in Figures 1b and 1c, respectively).

As we want to discover the topics that are being discussed in the real-time stream of messages that we receive, we develop an online algorithm to cluster the incoming tweets on an hourly basis. To test this algorithm, we select two intervals that span a total of two weeks, for which we compare the output of our algorithm for a given set of tweets to the results of a clustering algorithm run on the entire set of tweets at once. Due to the down-time of the scraper and the associated loss of tweets, we choose April 1st to April 15th and April 23th to May 7th as the intervals we process.

IV. CLUSTERING

As we are interested in topics that are discussed with respect to a large quantity of products, we use a post-hoc clustering technique on all tweets mentioning a certain product in an interval to produce a 'ground-truth' or baseline of topics, instead of using labor-intensive human annotated datasets. Since we do not know how many clusters there are going to be, we choose a clustering algorithm that does not need

[1] https://www.groentenfruithuis.nl/
[2] http://www.floricode.com/

 ISBN: 978-1-61208-603-3

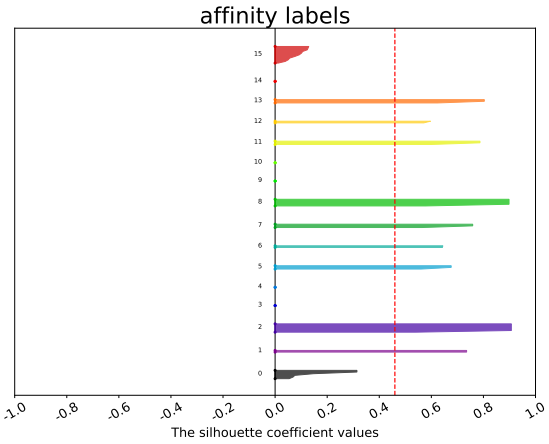


Figure 2. Silhouette scores for clusters of tweets mentioning 'celeriac' that were placed between April 23th and May 7th.

the number of clusters as an input parameter. Furthermore, we want to use an off-the-shelf clustering algorithm that is contained in an existing and actively maintained python module. Therefore, we choose to use Affinity Propagation [8] to produce our baseline, which is contained in the *scikit-learn* [19] module. Furthermore, we employ silhouettes [20] on the clusters formed by Affinity Propagation, to measure their consistency. If a silhouette score of a member of a cluster is close to 1 it is clustered correctly. Silhouette scores close to -1 indicate that the element is similar to different clusters. Clusters that consist of a single element automatically receive the score 0. As an example of this analysis, Figure 2 shows the silhouette scores for the tweets mentioning 'celeriac'. The average silhouette score of all elements is indicated by the red dashed line. The silhouette plot visualizes the silhouette scores per item that has been clustered. Each cluster is represented by a different color and the larger the silhouette is, the more tweets are contained in that cluster.

V. Obtaining Long-Term Topics

Although the post-hoc clustering approach works quite well using data from a longer interval, applying the same approach in a real-time fashion is not possible, since this algorithm can only process a complete dataset. Therefore, we develop an approach that can combine the results of hourly clustering, based on the received tweets during that hour.

This approach employs three phases. For the first phase, the tweets mentioning a certain product over the last hour are combined into a corpus. In this step, the tweets are tokenized, stemmed and Part Of Speech (POS) tagged. Then, using an N-dimensional space that represents this constructed corpus per product, the tweets are clustered using Affinity Propagation [8]. After this step, the clusters are represented by the set of tokens contained in their corresponding tweets. Finally, all new clusters are compared to clusters that have been found in previous hours through what we define as a 'stories'. Such a story can be seen as a cluster of clusters over time, and thereby combines the tweets that are similar. The algorithm we use to combine a list of clusters, denoted by C,

Require: list of current stories: S, list of clusters C, similarity threshold: J_t, original similarity threshold: J_o and maximum idle time m_i.

for $c \in C$ **do**
  boolean matched = False
  for $s \in S$ **do**
    if s is similar to c: **then**
      matched = True
      add cluster c to story s
    end if
  end for
  if not matched **then**
    add c to new story s' and add s' to S
  end if
end for
for $s \in S$ **do**
  update delay: $d_s = d_s + 1$
  if $d = m_i$ **then**
    close s and remove s from S.
  end if
end for
return S

Figure 3. *Storify* algorithm that assigns the clusters of to the stories.

with the list of current stories, denoted by S, is described in the algorithm displayed in Figure 3. Besides the input of the clusters and stories, the algorithm uses three other parameters. The first two parameters, the similarity threshold and the original threshold, denoted by J_t and J_o, respectively, are used to determine whether a cluster and a story are similar to each other. When a cluster is compared to a story, the comparison is done on two levels. First, the similarity between the tokens of the cluster and the current tokens of the story are calculated. Secondly, the similarity between the cluster tokens and the original story tokens are calculated. Both these similarities are calculated using the Jaccard similarity [21]. The Jaccard similarity between two sets A and B is defined as

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

If both values are not below their respective thresholds, being J_t and J_o, the cluster is added to the story and its tokens become the current story tokens. The reason that we employ two thresholds for the question of similarity is to ensure that the topic of the story does not drift over time from one topic to another. Finally, the maximum idle time, denoted by m_i, is defined as the maximum time that a story remains active without having a cluster added to it. When a story has not obtained an additional cluster for m_i time intervals, then it is closed. We do this to ensure that topics are not dragged on too long, without adding interesting messages to them.

Suppose clustering is done using all incoming tweets during the last hour. As an example of how the algorithm works, consider the following example tweets *"Pick tulips on Dam Square, for free!"*, placed on January 21st 2017 around 9:30 AM, and *"Tulipday on Dam Square great succes, 20.000 free tulips picked."*, placed on January 22nd 2017 around 2:15 PM, which have both been retweeted twice within a few minutes after posting. Then, one of the clusters on January

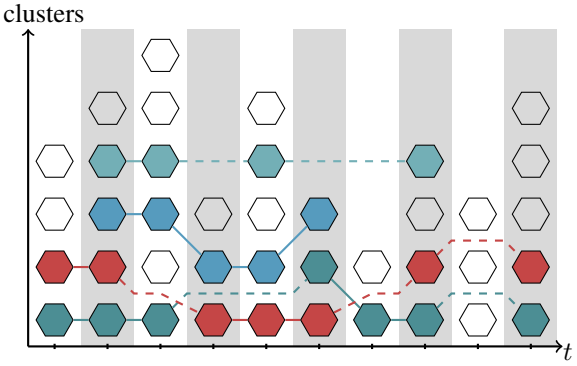


Figure 4. Example of clusters that are linked through time by the storify algorithm.

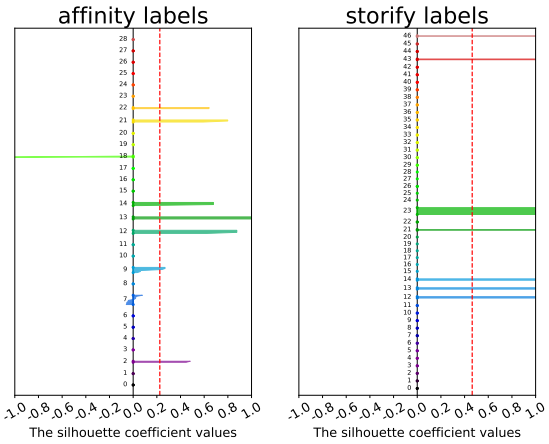


Figure 5. Silhouette scores for clusters and stories of tweets mentioning 'chrysantemum', placed between April 23th and May 7th.

21st at 10 AM will contain the first tweet and its retweets, containing the following tokens $[pick, tulip, Dam\ Square, free]$. Suppose this cluster triggers a new story, then this story will match with the cluster of the messages places on January 22nd 2017 (the tokens of this cluster are: $[Tulipday, Dam\ Square, succes, tulip, free, pick]$, thus the story and cluster are similar) and if it has not been more than m_i hours ago since the last cluster was added, the new cluster is added to the story.

Figure 4 visualizes an example of our approach to combine clusters through time, which we call the *storify* algorithm. On the y-axis, individual clusters are indicated by black hexagons and the x-axis shows how time progresses. The connections between clusters over time are indicated by filling the clusters in the color of the overal story (e.g., red). If two clusters are combined in the same story in consecutive hours, they are linked by a solid line. If there are a few hours in which a story has been idle, this is indicated by a dashed line.

VI. Performance Measurement

Measuring the performance of the storify algorithm can pose a challenge, because it is difficult to determine a ground-truth for all products considered. Therefore, we employed a post-hoc clustering algorithm on the selected intervals of tweets mentioning a product. Also, we execute the storify algorithm on the same tweets, where we set the parameters $m_i = 48$, $J_\mathrm{t} = 0.9$, and $J_\mathrm{o} = 0.6$. These parameter values are chosen with the intention to only cluster very similar tweets and to join clusters in a story if they mention similar terms. Futhermore, the max idle time of two days ensures that we do not exclude intervals that arise through the natural circadian rhythm of use Twitter.

Let us first consider the outcomes of both approaches for a single product. Here, we again use silhouettes [20] to visualize the how well the tweets are divided up into stories for our algorithm and clusters for the Affinity Propagation algorithm. Recall that if a silhouette score of a member of a cluster is close to 1 it is clustered correctly. Also, scores close to -1 indicate that the element is similar to different clusters. Figure 5 gives an example of the silhouette scores for tweets mentioning 'chrysanthemum' in the interval April 23th and May 7th. In this figure, we see that the average silhouette score for the storify clustering is larger than the Affinity Propagation clustering. Furthermore, in the Affinity plot, we see that some tweets have a negative silhouette score, whereas in the storify plot all values are positive. Even though the results are not consistent over all products, our approach outperforms the naive total clustering and provides for a better fit of the data for the product 'chrysantemum'.

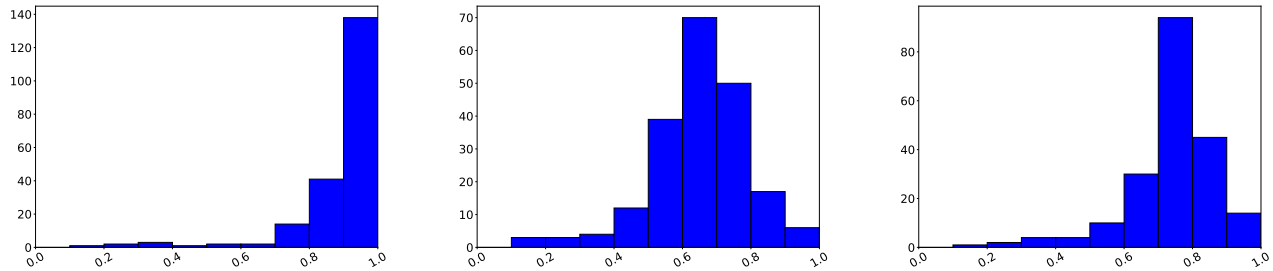
For a more general comparison of both approaches, we use three metrics used for cluster comparison, namely homogeneity, completeness, and the v-measure. Homogeneity measures if each cluster contains only members of a single class. Completeness measures if all members of a given class are assigned to the same cluster. Finally, the v-measure is defined as the harmonic mean of homogeneity and completeness of the clustering, as defined in [22].

Figure 6 shows the distribution of the three parameters for the described parameter settings. Clearly, the completeness scores are the lowest overall. This can be easily explained, since the storify approach gives more clusters of singular tweets than the Affinity Propagation does. This is a direct result of the usage of the maximum idle time and is, therefore, an expected outcome. In general, these three metrics are all skewed towards 1. Therefore, we can conclude our approach gives similar results as the naive total clustering approach, even though we have not optimized the parameters of our model.

VII. Conclusion and Future Work

Given the results of our analysis thus far, the storify algorithm appears to be very promising for application to a real-time social media feed. Even though we have not yet optimized the parameters of the model, the results compare very well with a direct post-hoc clustering done on all the data. The only difference between the two approaches is that the overall clustering finds fewer clusters containing a single tweet. To assess whether this difference undermines the validity of the algorithm, we plan to extend the analysis of these results not only to the groups in which the tweets are clustered, but also to the time at which these tweets are placed. For this analysis, the metric proposed by Krippendorff [23] seems very promising.

 ISBN: 978-1-61208-603-3



(a) Homogeinity (b) Completeness (c) V-Score

Figure 6. Distribution of homogeneity, completeness, and v-score based on the outcomes over all products, based on tweets placed from April 23th to May 7th.

Using this metric, we can also compare whether or not the time of placement of the tweets that are clustered between approaches, are similar or very different. Furthermore, we use a one-to-one mapping to map clusters to stories. Using a one-to-many mapping gives the opportunity for stories to become overlapping, which is an interesting topic for further study.

At this moment, the algorithm only runs ad-hoc using data acquired in our tool the HortiRadar [24]. Given the promising results and our interest to find topics that are discussed in a real-time feed, we aim to implement the algorithm in the HortiRadar. Using this real-time implementation, we can then show a visualization of the stories identified by the algorithm in the HortiRadar, which makes the identification of stories in a real-time feed a lot easier. Simultaniously, this visualization can be used as a validation of the chosen parameter settings and the clustering mechanism. Once the real-time visualization is up and running, the next step is to use the results of this study for business purposes in the horticulture industry.

ACKNOWLEDGMENTS

This work is funded as part of PPS KV 1406-101 of the Topsector Tuinbouw & Uitgangsmaterialen. The author thanks Sandjai Bhulai and Rahiel Kasim for their feedback and help designing and implementing this approach.

REFERENCES

[1] J. G. Fiscus and G. R. Doddington, *Topic Detection and Tracking Evaluation Overview*. Boston, MA: Springer US, 2002, pp. 17–31.

[2] B. Pouliquen, R. Steinberger, and O. Deguernel, "Story tracking: Linking similar news over time and across languages," in *Proceedings of the Workshop on Multi-source Multilingual Information Extraction and Summarization*, ser. MMIES '08. Stroudsburg, PA, USA: Association for Computational Linguistics, 2008, pp. 49–56.

[3] S. Bhulai, *et al.*, "Trend Visualization on Twitter: What's Hot and What's Not?" *IARIA DATA ANALYTICS*, pp. 43–48, 2012.

[4] S. Phuvipadawat and T. Murata, "Breaking news detection and tracking in twitter," in *2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, vol. 3, Aug 2010, pp. 120–123.

[5] J. Allan, *Introduction to Topic Detection and Tracking*. Boston, MA: Springer US, 2002, pp. 1–16.

[6] J. Weng and B.-S. Lee, "Event detection in Twitter," *ICWSM*, vol. 11, pp. 401–408, 2011.

[7] X. Huang, X. Zhang, Y. Ye, S. Deng, and X. Li, "A topic detection approach through hierarchical clustering on concept graph," *Applied Mathematics & Information Sciences*, vol. 7, no. 6, p. 2285, 2013.

[8] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.

[9] B. Mukhoty and R. Gupta, "A parameter-free affinity based clustering," *CoRR*, vol. abs/1507.05409, 2015.

[10] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[11] M. Ester, *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[12] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[13] F. Holz and S. Teresniak, "Towards automatic detection and tracking of topic change," *Computational linguistics and intelligent text processing*, pp. 327–339, 2010.

[14] X. Wang and A. McCallum, "Topics over time: A non-markov continuous-time model of topical trends," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '06. New York, NY, USA: ACM, 2006, pp. 424–433.

[15] R. Swan and J. Allan, "Automatic generation of overview timelines," in *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '00. New York, NY, USA: ACM, 2000, pp. 49–56.

[16] Twitter realtime filtering. Retrieved: September 30, 2017. [Online]. Available: https://developer.twitter.com/en/docs/tweets/filter-realtime/overview

[17] Twitter rate-limits. Retrieved: September 30, 2017. [Online]. Available: https://developer.twitter.com/en/docs/basics/rate-limits

[18] M. ten Thij, S. Bhulai, W. van den Berg, and H. Zwinkels, "Twitter Analytics for the Horticulture Industry," *IARIA DATA ANALYTICS*, pp. 75–79, 2016.

[19] Scikit-learn python module. Retrieved: September 30, 2017. [Online]. Available: http://scikit-learn.org/stable/

[20] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53 – 65, 1987.

[21] P. Jaccard, "Distribution of alpine flora in the dranses basin and in some neighboring regions," *Bull. Soc. Vaud. Sci. Nat.*, vol. 37, pp. 241–272, 1901.

[22] A. Rosenberg and J. Hirschberg, "V-measure: A conditional entropy-based external cluster evaluation measure," in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning(EMNLP-CoNLL)*, 2007, pp. 410–420.

[23] K. Krippendorff, "On the reliability of unitizing continuous data," *Sociological Methodology*, vol. 25, pp. 47–76, 1995.

[24] Hortiradar. Retrieved: September 30, 2017. [Online]. Available: https://hortiradar.bigtu.nl/hortiradar/

 ISBN: 978-1-61208-603-3