A Pipeline for Multimedia Twitter Analysis through Graph Databases: Preliminary Results

Roberto Boselli[1,3], Mirko Cesarini[1,3], Fabio Mercorio[1,3], Mario Mezzanzanica[1,3] and Alessandro Vaccarino[2]

[1]*Dep. of Statistics and Quantitative Methods, University of Milan-Bicocca, 8 via Bicocca degli Arcimboldi, Milan, Italy*

[2]*Aubay Italia SpA, 2 Largo la Foppa, Milan, Italy*

[3]*CRISP Research Centre - University of Milan-Bicocca, Milan, Italy*

Keywords: Graph-Database, Social Network Analysis, Microblogging.

Abstract: Twitter is a microblogging service where users post not only short messages, but also images and other multimedia contents. Twitter can be used for analyzing people public discussions, as a huge amount of messages are continuously broadcasted by users. Analysis have usually focused on the textual part of messages, but the non-negligible number of images exchanged calls for specific attention. In this paper we describe how the tweet multimedia contents can be turned into a knowledge graph and then used for analyzing the messages sent during marketing campaigns. The information extraction and processing pipeline is built on top of off-the-shelf APIs and products while the obtained knowledge is modelled through a Graph Database. The resulting knowledge graph was useful to explore and identify similarities among different marketing campaigns carried out using Twitter, providing some preliminary but promising results.

1 INTRODUCTION AND MOTIVATION

On the Twitter microblogging service, users post millions of short messages daily, often including pictures and videos. The number of tweets containing multimedia contents is rapidly growing, mainly due to the diffusion of several services that allow users to spread contents over Twitter *directly* from other external social media platforms e.g., Facebook, LinkedIn, and Instagram.

In such a context, we believe images related to tweet's texts can be effectively used to better investigate the tweet's informative content by extracting and recognising common features within pictures, such as emotional face attributes, close-up people, image specific colours, objects, brand logos, etc. This, in turn, would contribute to understand how images attached to tweets are used in marketing campaigns, identifying relationships among image features in different social campaigns too.

Some existing approaches are aimed at extracting images from Twitter related to some real-life events e.g., (Kaneko and Yanai, 2016), as well as to visualise them using Twitter geo-tags (Yanai et al., 2014). On the other side, several studies use SNA on social media data, in particular Twitter data, to analyze user behaviours and interactions in several contexts, ranging from marketing to social community management (Java et al., 2007; Ediger et al., 2010; Cheong and Cheong, 2011).

Differently, our approach is aimed at building a knowledge-graph of tweets' photo features. Then, we query the resulting graph database for identifying common patterns within tweets, to analyse how a brand marketing campaign has been conducted.

Indeed, Social Media Marketing (SMM) is the evolution of traditional marketing process where brands look for both visibility and dialogue with consumers using social media. SMM allows brands to interact with customers more directly and equally, e.g. through a stream of tweets in Twitter or comments in Facebook. Interactions and comments generate the so-called "engagement", which allows brands to get feedback, opinions, advice, review etc. (Hoffman and Fodor, 2010). SMM offers to consumers the opportunity to express themselves without intermediaries, while it allows brands to listen and meet the customer's needs, and let them also get involved in the marketing projects (crowdsourcing). The mes-

Boselli, R., Cesarini, M., Mercorio, F., Mezzanzanica, M. and Vaccarino, A.
A Pipeline for Multimedia Twitter Analysis through Graph Databases: Preliminary Results.
DOI: 10.5220/0006490703430349
In *Proceedings of the 6th International Conference on Data Science, Technology and Applications (DATA 2017)*, pages 343-349
ISBN: 978-989-758-255-4

sages exchanged via Twitter offer companies new ideas and insights to promote web marketing campaigns and to improve consumer relationships. Each activity on social media must be measured with the right Social Media Analytic metrics according to specific analytical objectives e.g., the sentiment analysis or the engagement rate calculation (Peters et al., 2013). Brands are constantly looking on new ways to extract information from Twitter messages and Social Media interactions in general.

From a practical perspective, graph-based applications have yet proved their powerfulness in real-life application, such as social network analyses (Lin et al., 2012; Dries et al., 2009; Zou et al., 2009; Amato et al., 2017; Appel and Moyano, 2017), data cleaning (Boselli et al., 2013; Mezzanzanica et al., 2013; Mezzanzanica et al., 2015a), biology (Eckman and Brown, 2006; Benhiba et al., 2017), Web mining (Schenker et al., 2005), graph-exploration (Della Penna et al., 2010; Mercorio, 2013) and semantic Web (Hayes and Gutierrez, 2004; Zeng et al., 2013), healthcare (Boselli et al., 2014), just to cite a few.

Paper's Goal. In this paper we present a system that automatically retrieves tweets and builds a knowledge graph about the attached multimedia contents for analysis purposes. Concisely, the system collects tweets and attached images sent from a given list of user accounts, then it employs off-the-shelf machine-learning and vision APIs to identify the images main elements and features. Finally, it builds a knowledge graph to perform SNAs and graph-traversal queries as well. Our framework has been designed, implemented, and then tested focusing on the use-case scenario of a marketing campaign performed by four well-known fast-food restaurants. We also provide some preliminary but promising results.

2 Graph-DB AT A GLANCE

In recent years the well-known NoSQL movement brought to the fore new data model paradigms that differ significantly with respect to the classical relational model (e.g., key-values, document databases, column-oriented and graph-db). For further details on NoSQL databases the interested reader can refer to (Han et al., 2011; Cattell, 2011).

All these new paradigms share some interesting features compared with classical relational model (see, e.g. (Stonebraker, 2010)), such as a flexible schema that can evolve to always fit the data, the ability to horizontally scale with ease as well as the native support for sharing. However, most of NoSQL databases store sets of disconnected documents and values (aka *aggregate models*), as in the case of key-value and document databases. This in turn makes it difficult to use them for connected data. Differently, graph databases (see, (Angles and Gutierrez, 2008)) present three distinct characteristics useful for our purposes: (i) they allow for a natural modelling of the data, through nodes and edges between nodes. Each node might contain attributes in a schema-free fashion while edges enable the possibility to connect entities with labels and properties; (ii) queries can be performed directly on the graph, thus inheriting a huge amount of efficient algorithms and well-formalized problems on graphs (e.g., shortest-path, A*, SNA metrics, etc.); (iii) in recent years, graph-databases are growing in importance in both industrial and academic communities, making available a number of (stable enough) solutions for storing and querying graphs in real-life situations (e.g., Neo4j (Webber, 2012), OrientDB (Tesoriero, 2013), Titan (Titan, 2017), just to cite a few).

Formally speaking, a graph is a pair $G = (V, E)$ where V is a finite set of *nodes* and E a set of unordered pairs $(u, v) \in E$, the so-called *edges*. Two nodes u and v are adjacent if the pair $(u, v) \in E$. Although a graph-structure is simple and well-known, a graph-store can be implemented in several ways, as the case of Neo4j. Neo4j is a property graph database, this means it is an attributed, labelled, directed multi-graph[1]. Neo4j is composed of four building blocks:

Labels associate a common name to a set of Nodes or Relationships to allow for fast indexing. From a conceptual perspective, labels can be seen as a construct to model the E/R model hierarchies, as a node/relation can have more than one label at a time.

Nodes represent a tuple in a common relational database, each of which can vary in length having different data types;

Relationships can exist between nodes. In a property graph model we can represent directed binary relations, that always have a start and end node.

Properties are key-values pair that can be included into any nodes or attached to any relationships.

[1] A multi-graph is a graph where multiple edges between two nodes are permitted and might be specified through labels.

3 THE PIPELINE ARCHITECTURE

All the process is designed to grant versatility while analysing different domains and scenarios.
The whole architecture has been scheduled through the ETL Talend (Bowen, 2012) orchestration component that allows to design, schedule and monitor each phase of the system developed. Talend layer is also used to handle JSON interface files between different phases of the process.
A Python layer is responsible of interfacing with the API of Twitter and Google Cloud Vision , and to interact with Neo4j.

In Fig. 1 we show both the architecture and the pipeline workflow, the latter mainly composed of 4 phases, namely:

Figure 1: System Architecture.

1. **Phase. Collecting Tweets.** Public Twitter APIs have been employed via Python calls to collect tweets using as input (1) a specific user account of interest, and (2) a list of keywords. A JSON file is returned for each tweet retrieved[2]. The dataset includes, among other, some information relevant for our purposes, namely: (i) *tweet data*, that include tweet ID, text, date and time of publication, geolocation and retweet details; (ii) *user data*, such as ID, followers count and location; (iii) the *hashtag list* and (iv) the url list of the attached *tweet images* (if any).
   All these information are stored locally through TALEND for being used by the next phase.

2. **Phase. Downloading Images.** The pipeline scans each url attached to a tweet, and downloads the corresponding image. Clearly, if several tweets contain the same image file (i.e., the tweet is a retweet) the image is downloaded only once.

3. Phase. Image Processing. All the images collected at the previous step are stored on a cloud service for easily interact with Google's Cloud Vision platform through proprietary REST API, called Google Cloud Vision API. The latter performs an image content analysis to automatically recognise specific items, such as objects, faces, known logos, text, colours, and the sentiments related to the photo too. According to the official documentation[3] the Google service employs machine learning algorithms for classifying, among other:
   - the *close-up entity* labelled, that is computed from a wide range of object categories (e.g., car, dog, face, etc.). A *confidence score* is also attached to each entity recognised within the image;
   - the *OCR recognition* is able to retrieve texts (automatically detecting the language) and brand logos;
   - the *properties detection* feature allows identifying the set of characteristics of a picture, such as its dominant colour in RGB format;
   - the *facial detection* feature can detect whether faces appear in images or not, and a set of eight emotional facial attributes of the identified people like joy, sorrow, and anger.

4. **Phase. Building the Knowledge Graph.** All the information about tweets (gathered in Phase 1) along with the image features collected from Phase 2 and 3, are arranged into a *graph-database model*. For the sake of completeness here we report both the E/R model and the Graph-DB model respectively in Fig. 2 and Fig. 3. As one might note, entities of the E/R corresponds to labels in the property graph models, whilst relationships are modelled as edges.

4 PRELIMINARY EXPERIMENTAL RESULTS

In this preliminary experimental phase we analysed the images referenced by 1,000 tweets sent from four distinct burger restaurant accounts, namely: *Mc Donald's*, *Burger King*, *Taco Bell* and *KFC*. In Tab. 1 we report the statistics of the resulting knowledge graph as modelled in Neo4j.

The goal of this experimental evaluation was to assess the effectiveness of the proposed approach in a sandbox environment.

[2]The interested reader can refer to the official Twitter API documentation for further details at https://dev.twitter.com/rest/public/search

[3]https://cloud.google.com/vision

Table 1: Graph statistics.

Nodes							Edges						
Photo	Tweet	User	HashTag	Color	Label	FaceEmotion	:Contains	:Face_Emotion	:Attach	:Has	:Post	:Color	:Jaccard
710	940	841	727	450	587	7	4,640	1,021	945	2,183	940	2,501	6,010

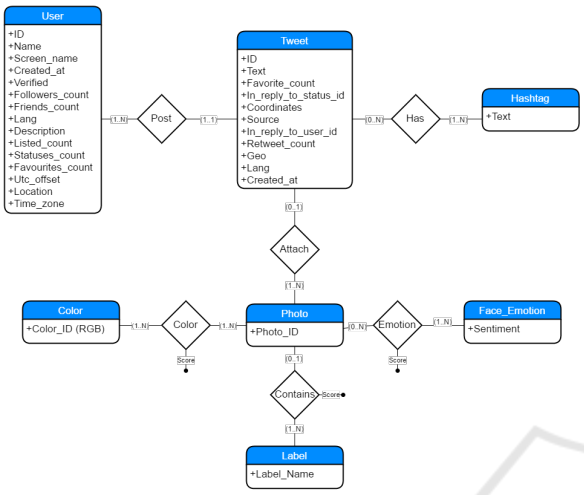


Figure 2: Data Model of the Knowledge extracted from the twitted multimedia content.

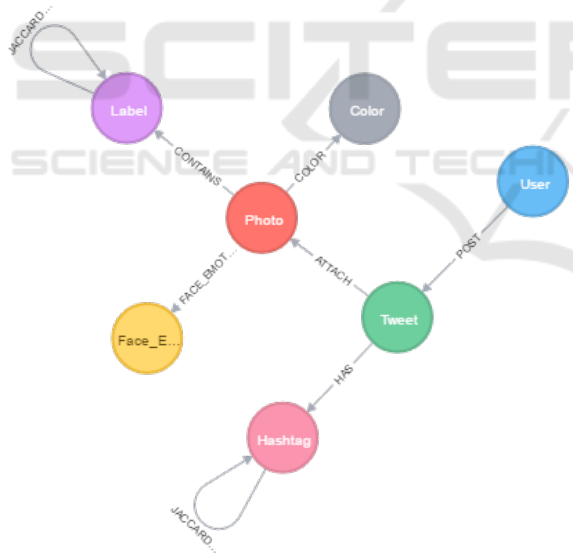


Figure 3: The System Data Model.

Computing Jaccard Similarities. Analysing the Graph DB obtained using the pipeline described in Sec. 3, we computed the Jaccard Index (Real and Vargas, 1996) (as a similarity relationship) on pairs of Label nodes (Cypher Query 1). It is worth recalling that label nodes represent elements found within images by the Google Vision API. The Jaccard Index between two labels is computed as the number of images having both labels over the number of images that have at least one of the elements. Let A be the set of images having element a and let B the set of images having element b, the Jaccard Index is computed as

$$J_{a,b} = |A \cap B| / |A \cup B| \quad (1)$$

This analysis allows one to identify recurrent elements within the photos published by a vendor and to derive insights about the marketing plan adopted. By looking ad the graph in Fig. 5(a) it is evident that McDonald's focuses more on *outdoor* images with respect to the competitors. At the same time, Burger King focuses more on sports related images as outlined in Fig. 5(b). This analysis is useful to understand the features that brands use in their marketing campaign and how these features are related as well.

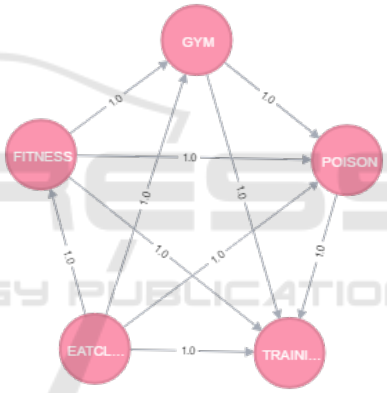


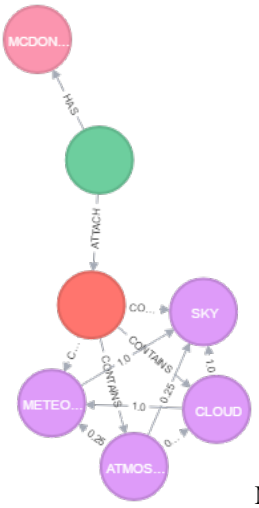
Figure 4: Example of a high related Clique.

Local Clustering Coefficient. (Watts and Strogatz, 1998) is a measure that allows computing the degree to which nodes in a graph tend to cluster together (aka transitivity coefficient). A Local Clustering Coefficient was computed above the Jaccard Index over hashtags to detect hashtags that are always used together, and their identification is valuable since it allows analysts to discover interesting relationships among the topics dealt in tweets. Let $u \in V(G)$ a node of the network, the local clustering coefficient for u is an index defined as

$$lc_u = \frac{|\{e_{vw} \in E : e_{uv} \in E, e_{uw} \in E\}|}{n_u(n_u - 1)} \quad (2)$$

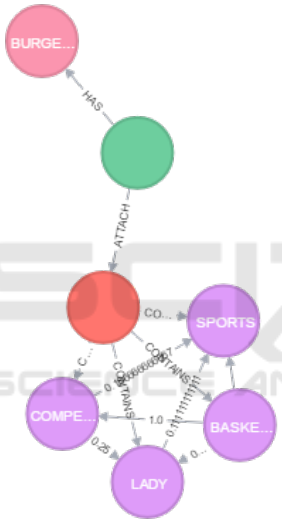
that is the ratio between the number of triangles (3-cliques) and the maximum number of triangles in which the node could be involved, that depends on the number of nodes in the neighbourhood of u, namely

$$n_u = |\{v_j : e_{ij} \in E \vee e_{ji} \in E\}| \quad (3)$$



Mc Donald's

(a)



Burger King's

(b)

Figure 5: Mc Donald's and Burger King's Jaccard Indexes.

The Jaccard Index was computed also among hashtags. Let H be the set of tweets having the hashtag h and let K the set of tweets having the hashtag k, the Jaccard Index between the hashtags k and h is

$$J_{h,k} = |H \cap K| / |H \cup K| \quad (4)$$

In Fig. 4 an example of Clique is showed where the Jaccard Index is equal to 1 among all the involved hashtags while the Cypher Query 2 shows the code used for computing the local clustering coefficient for each hashtag pairs.

Finally, we can also analyse features relevance for each vendor, as we show in Fig. 6. This allowed us to have a bird-eye-view of the main features that each vendor exploited in its campaign.

The just outlined analyses are an example of the insights that can be identified analysing twitter contents and images using a knowledge graph. Furthermore, the pipeline presented in this paper is general enough to be used in different scenarios too.

Cypher Query 1: Compute Jaccard Similarity between labels

```
MATCH (p:Photo)-[r]-(l:Label)
WHERE l.Label_Name = X
WITH COUNT(distinct r) AS X_lab_counter
MATCH (p:Photo)-[r]-(l:Label)
WHERE l.Label_Name = Y
WITH X_lab_counter, COUNT(distinct r) AS Y_lab_counter
MATCH (l1:Label)-[]-(p:Photo)-[]-(l2:Label)
WHERE l1.Label_Name = X AND l2.Label_Name = Y
WITH X_lab_counter, Y_lab_counter, COUNT(distinct p) AS
     counter3
WITH (X_lab_counter+Y_lab_counter)-counter3 AS Unione
MATCH (l1:Label)-[]-(p:Photo)-[]-(l2:Label)
WHERE l1.Label_Name = X AND l2.Label_Name = Y
WITH Union, COUNT(distinct p) AS Intersect
RETURN (Intersect*1.0/Union) AS Jaccard
```

Cypher Query 2: Local Clustering Coefficient assignment and Clique detection

```
MATCH (a:Hashtag)-[:JACCARD_LINK_HASHTAG]-(b:Hashtag)
WITH a, collect(b) as sn, count(distinct b) as n,(count(
     distinct b)*(count(distinct b)-1))/2 as nk
MATCH (a)-[:JACCARD_LINK_HASHTAG]-(b1)-[rel:
     JACCARD_LINK_HASHTAG]-(b2)-[:JACCARD_LINK_HASHTAG
     ]-(a)
WITH a, sn, n, nk, count(distinct rel) as r, toFloat(
     count(distinct rel))/toFloat(nk) as coef
WHERE coef = 1
WITH a, sn
FOREACH(c in RANGE(0, size(sn)-1) | FOREACH(n1 in [sn[c
     ]] | Set n1.CliqueId= id(a),a.CliqueId=id(a)))

MATCH (a:Hashtag) RETURN distinct a.CliqueId, count(a)
     as CliqueDim ORDER BY CliqueDim desc
```

5 CONCLUSIONS AND EXPECTED OUTCOMES

Traditionally, analyses on Twitter messages have focused on textual contents while attached images have been hardly considered. Nevertheless, extracting information over the pictures exchanged via twitter is a rich source of valuable data and meaningful analysis can be performed thereon. Nowadays, off-the-shelf image processing APIs can extract several interesting information from pictures e.g., dominant colours and logos, people faces, and emotional facial attributes like joy, sorrow, and anger.

The data extracted from images can be coupled with other (more traditional) information extracted from twitter messages (e.g., the relationships among users, exchanged messages, and hashtags). A knowledge graph can be built thereon and then used for subsequent analysis.

The information extraction and analysis pipeline described in this paper was used to investigate

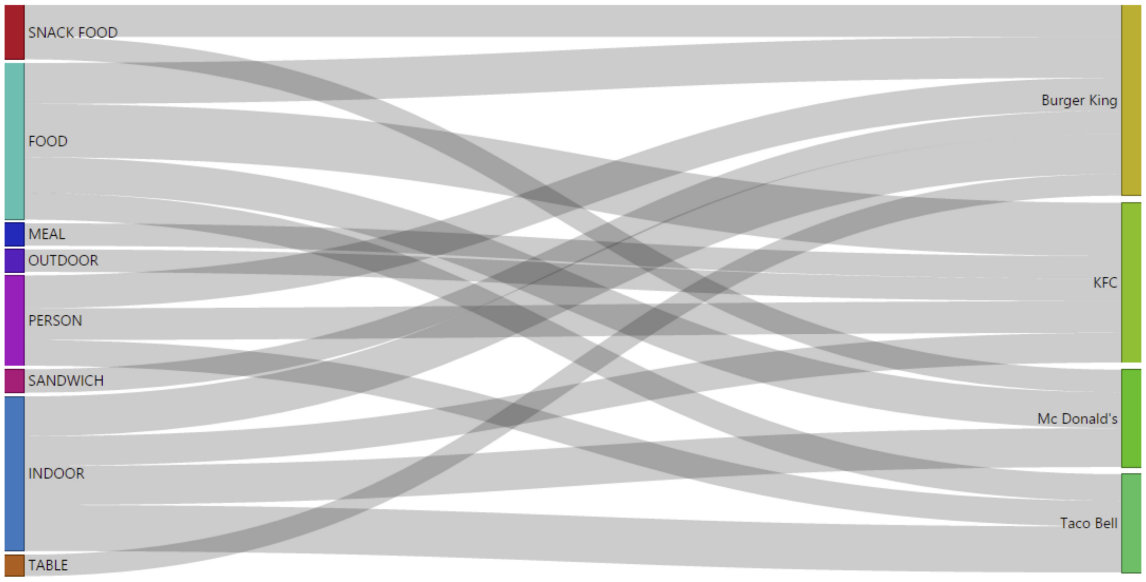


Figure 6: Correlation between Photo tag & Vendor (confidence higher than 97%).

the marketing campaigns performed by four burger restaurants on a small but significant dataset. The graph database that hosted the knowledge graph has proved to be suitable for exploring and identifying interesting relationships among the images used for each campaign.

Images are very frequently attached to Twitter messages and they convey some additional information that are not present in text messages. The information extraction pipeline and the knowledge graph presented is general enough to be used for several twitter analysis tasks, non limited to the specific case presented in this paper. We are now working on applying this approach to a wide number of Tweets in different marketing campaigns, implementing the pipeline over a big data architecture to scale-out effectively. Furthermore, we also intend to build a graph-based model for reasoning with labour market data, both structured (Mezzanzanica et al., 2011; Mezzanzanica et al., 2015b) and unstructured (see, e.g., (Amato et al., 2015)) that would allow implementing graph-traversal queries and SNA metrics as well.

ACKNOWLEDGMENT

The authors would like to thank Dr. Carla Marini for her invaluable assistance and thoughtful discussions while working on this project.

REFERENCES

Amato, F., Boselli, R., Cesarini, M., Mercorio, F., Mezzanzanica, M., Moscato, V., Persia, F., and Picariello, A. (2015). Challenge: Processing web texts for classifying job offers. In *Semantic Computing (ICSC), 2015 IEEE International Conference on*, pages 460–463.

Amato, F., Moscato, V., Picariello, A., and Sperlí, G. (2017). *Influence Maximization in Social Media Networks Using Hypergraphs*, pages 207–221. Springer International Publishing, Cham.

Angles, R. and Gutierrez, C. (2008). Survey of graph database models. *ACM Computing Surveys (CSUR)*, 40(1):1.

Appel, A. P. and Moyano, L. G. (2017). Link and graph mining in the big data era. In *Handbook of Big Data Technologies*, pages 583–616. Springer.

Benhiba, L., Loutfi, A., and Idrissi, M. A. J. (2017). A classification of healthcare social network analysis applications. *BIOSTEC 2017*, page 147.

Boselli, R., Cesarini, M., Mercorio, F., and Mezzanzanica, M. (2013). Inconsistency knowledge discovery for longitudinal data management: A model-based approach. In *SouthCHI13 special session on Human-Computer Interaction & Knowledge Discovery, Lecture Notes in Computer Science, vol. 7947*. Springer.

Boselli, R., Cesarini, M., Mercorio, F., and Mezzanzanica, M. (2014). *A Policy-Based Cleansing and Integration Framework for Labour and Healthcare Data*, pages 141–168. Springer Berlin Heidelberg, Berlin, Heidelberg.

Bowen, J. (2012). *Getting Started with Talend Open Studio for Data Integration*. Packt Publishing Ltd.

Cattell, R. (2011). Scalable sql and nosql data stores. *Acm Sigmod Record*, 39(4):12–27.

Cheong, F. and Cheong, C. (2011). Social media data mining: A social network analysis of tweets during the australian 2010-2011 floods. In Seddon, P. B. and Gregor, S., editors, *15th Pacific Asia Conference on Information Systems (PACIS)*, pages 1–16. Queensland University of Technology.

Della Penna, G., Intrigila, B., Magazzeni, D., and Mercorio, F. (2010). A PDDL+ benchmark problem: The batch chemical plant. In *Proceedings of the The 20th International Conference on Automated Planning and Scheduling (ICAPS 2010)*, pages 222–225, Toronto, Canada. AAAI Press.

Dries, A., Nijssen, S., and De Raedt, L. (2009). A query language for analyzing networks. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 485–494. ACM.

Eckman, B. A. and Brown, P. G. (2006). Graph data management for molecular and cell biology. *IBM journal of research and development*, 50(6):545–560.

Ediger, D., Jiang, K., Riedy, J., Bader, D. A., Corley, C., Farber, R. M., and Reynolds, W. N. (2010). Massive social network analysis: Mining twitter for social good. In *39th International Conference on Parallel Processing*, pages 583–593.

Han, J., Haihong, E., Le, G., and Du, J. (2011). Survey on nosql database. In *Pervasive computing and applications (ICPCA), 2011 6th international conference on*, pages 363–366. IEEE.

Hayes, J. and Gutierrez, C. (2004). Bipartite graphs as intermediate model for rdf. In *International Semantic Web Conference*, pages 47–61. Springer.

Hoffman, D. L. and Fodor, M. (2010). Can you measure the roi of your social media marketing? *MIT Sloan Management Review*, 52(1):41.

Java, A., Song, X., Finin, T., and Tseng, B. (2007). Why we twitter: Understanding microblogging usage and communities. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis*, WebKDD/SNA-KDD '07, pages 56–65, New York, NY, USA. ACM.

Kaneko, T. and Yanai, K. (2016). Event photo mining from twitter using keyword bursts and image clustering. *Neurocomputing*, 172:143–158.

Lin, C.-Y., Wu, L., Wen, Z., Tong, H., Griffiths-Fisher, V., Shi, L., and Lubensky, D. (2012). Social network analysis in enterprise. *Proceedings of the IEEE*, 100(9):2759–2776.

Mercorio, F. (2013). Model checking for universal planning in deterministic and non-deterministic domains. *AI Communications*, 26(2):257–259.

Mezzanzanica, M., Boselli, R., Cesarini, M., and Mercorio, F. (2011). Data quality through model checking techniques. In Gama, J., Bradley, E., and Hollmén, J., editors, *Intelligent Data Analysis (IDA), Lecture Notes in Computer Science vol. 7014*, pages 270–281. Springer.

Mezzanzanica, M., Boselli, R., Cesarini, M., and Mercorio, F. (2013). Automatic synthesis of data cleansing activities. In Helfert, M., Francalanci, C., and Filipe, J., editors, *DATA 2013 - the International Conference on Data Technologies and Applications*, pages 138–149. SciTePress.

Mezzanzanica, M., Boselli, R., Cesarini, M., and Mercorio, F. (2015a). A model-based approach for developing data cleansing solutions. *Journal of Data and Information Quality (JDIQ)*, 5(4):13.

Mezzanzanica, M., Boselli, R., Cesarini, M., and Mercorio, F. (2015b). A model-based evaluation of data quality activities in KDD. *Information Processing & Management*, 51(2):144–166.

Peters, K., Chen, Y., Kaplan, A. M., Ognibeni, B., and Pauwels, K. (2013). Social media metrics. a framework and guidelines for managing social media. *Journal of interactive marketing*, 27(4):281–298.

Real, R. and Vargas, J. M. (1996). The probabilistic basis of jaccard's index of similarity. *Systematic biology*, 45(3):380–385.

Schenker, A., Kandel, A., Bunke, H., and Last, M. (2005). *Graph-theoretic techniques for web content mining*, volume 62. World Scientific.

Stonebraker, M. (2010). Sql databases v. nosql databases. *Communications of the ACM*, 53(4):10–11.

Tesoriero, C. (2013). *Getting Started with OrientDB*. Packt Publishing Ltd.

Titan (2017). Distributed graph database. http://titan.thinkaurelius.com/.

Watts, D. J. and Strogatz, S. H. (1998). Collective dynamics of small-worldnetworks. *nature*, 393(6684):440–442.

Webber, J. (2012). A programmatic introduction to neo4j. In *Proceedings of the 3rd annual conference on Systems, programming, and applications: software for humanity*, pages 217–218. ACM.

Yanai, K., Kaneko, T., and Kawano, Y. (2014). Real-time photo mining from the twitter stream: event photo discovery and food photo detection. In *Multimedia (ISM), 2014 IEEE International Symposium on*, pages 295–302. IEEE.

Zeng, K., Yang, J., Wang, H., Shao, B., and Wang, Z. (2013). A distributed graph engine for web scale rdf data. In *Proceedings of the VLDB Endowment*, volume 6, pages 265–276. VLDB Endowment.

Zou, L., Chen, L., and Özsu, M. T. (2009). Distance-join: Pattern match query in a large graph database. *Proceedings of the VLDB Endowment*, 2(1):886–897.