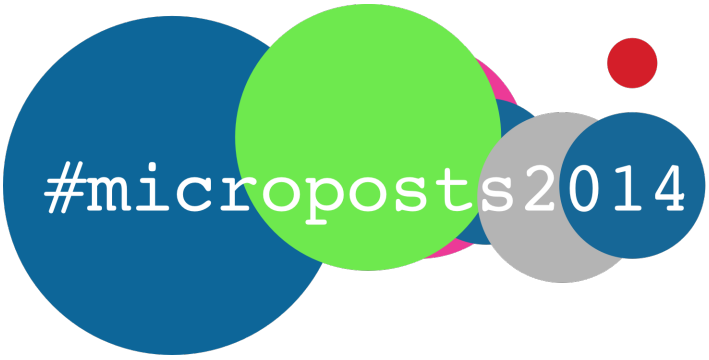
Proceedings of the 4th Workshop on

Making Sense of Microposts (#Microposts2014)

Big things come in small packages



at the 23rd International Conference on the World Wide Web (WWW'14)
Seoul, Korea
7th of April 2014

edited by
Matthew Rowe, Milan Stankovic, Aba-Sah Dadzie

Preface

The 4th Workshop on *Making Sense of Microposts (#Microposts2014)* was held in Seoul, Korea, on the 7th of April 2014, during the 23rd International Conference on the World Wide Web (WWW'14). #Microposts2014 sees a change in the workshop acronym from #MSM, to highlight our focus on *Microposts* – small chunks of information published online with minimal effort, via a variety of platforms and devices. The #Microposts journey started in 2011 at the 8th Extended Semantic Web Conference (ESWC 2011), then moved to WWW in 2012, where it has stayed, for the third year now.

The #Microposts series of workshops is unique in targeting researchers from a range of fields spanning both Computer Science and the Social Sciences. The aim is to harness the benefits different fields bring to research involving Microposts, and to maintain a focus on the end user and their interaction with other users and the physical and online worlds – the community who collectively publish this rich, varied information.

Microblogging platforms and other restricted size, text only and multi-media capable, instant communication tools are now so commonplace that applications are constantly being developed to enable their use not only on the desktop, but also on the go, from ordinary and smart phones, tablets and even public kiosks.While the well-known microblogging platforms – Twitter, Facebook, MySpace, Google+, Tumblr, Foursquare, Instagram and Pinterest, among others – cover a large portion of the online user base, other country and/or language-specific platforms such as Sina Weibo, and mobile-based messaging tools such as WhatsApp, are increasingly being used to share Micropost-type information. This medium of communication is no longer patronised predominantly by individual users sharing information informally within private networks and also with the wider public, but is used as mouth pieces by enterprise organisations and public bodies, to foster a feeling of more personal interaction with consumers and the wider, participating public. Disaster and emergency response and management, and political upheaval and crises, are two areas where social media has been shown to be particularly powerful for disseminating critical information to the individuals involved, and for broadcasting events as they occur to the outside world. Citizen reporters and scientists are now commonly accepted, or even anticipated, by organisations that traditionally relied mainly on trained experts. Microblogging and other social media platform usage is seen across all walks of life, from opinion mining and feedback solicitation for public consultations, to election campaigns and classroom participation.

With increasingly lower cost methods for publishing *Microposts* (often via mobile devices), and widespread use of informal and abbreviated language, the sheer scale and heterogeneity of Micropost data presents challenges for analysis, knowledge extraction and aggregation, further dissemination and reuse in any of a range of applications. At the same time, today's end user, understandably, has very high expectations for intuitive, minimal effort applications for tailored search and information retrieval across myriad, interconnected devices, customised to their current context – situation, location and proximity of others within their social and other networks, and influenced by unknown users with similar interests.

The #Microposts workshop was created to bring together researchers exploring novel methods for analysing Microposts, and for reusing the resulting collective knowledge extracted from such posts, both online and in the physical world. With each year we have seen novel, leading edge approaches to exploring this now ubiquitous, but still very valued, means of communication and the knowledge it generates. We are able to report wide interest in the workshop, with a good number of submissions from a range of fields in and across disciplines, mainly from Computer Science and the Social Sciences. Along with reports of applications in different domains, our contributors and audience have re-confirmed each year the importance of Microposts to the ordinary end user and, increasingly, public organisations and industry.
Many hearty thanks to all our contributors and participants. Submissions came from institutions all over the world – the main track saw authors from institutions across 11 different countries, and the challenge from 7. Interestingly, while challenges are often more popular with students, half the challenge submissions included authors from research institutions, including Microsoft Research, the Max Planck Institute, CNRS (France) and SAP Research.
Our Programme Committee are even more varied, coming from universities and research institutions around the world, as well as from industry, more than half of whom have reviewed for each of our four workshops. A very special thanks goes to each of them; their valued feedback resulted in a rich collection of papers and posters, each of which adds to the state of the art in leading edge research. We are confident that the #Microposts series of workshops will continue to foster a vibrant community, as we continue to work with the rich body of knowledge generated by the many and varied end users whose social and working lives span the physical and online worlds.

Matthew Rowe University of Lancaster, UK
Milan Stankovic Sépage / Université Paris-Sorbonne, France
Aba-Sah Dadzie The University of Birmingham, UK
#Microposts2014 Organising Committee, April 2014

Introduction to the Proceedings

The main workshop track attracted 12 submissions, 6 of which, all long papers, were accepted, along with a poster. These covered topics from machine learning, on Micropost classification and extraction, to data mining and analysis, and sentic and sentiment analysis. Applications were seen in incident and emergency response and management, and topic and opinion mining. We provide a brief introduction to these below.

The proceedings include the abstract of the keynote, '*Computational Social Science and Microblogs – The Good, the Bad and the Ugly*', presented by Markus Strohmaier of the Dept. of Computer Science, University of Koblenz-Landau, Germany.

Main Track Presentations

Micropost Mining and Analysis

Panisson *et al.,* in their paper *Mining Concurrent Topical Activity in Microblog Streams*, present a novel approach to topic mining from Twitter streams, in the context of recreating event timelines. Their evaluation, performed on a dataset sampled from the London 2012 Summer Olympics, shows a high degree of matching between the inferred timeline and the actual Olympics schedule.

Prapula G *et al.* introduce the notion of episode in the extraction of events from tweets, in *TEA: Episode Analytics on Short Messages*. Detection of *episodes* – significant moments when a particular entity gets traction on Twitter – constitute the basis for the application scenarios they present. Using data visualisation and social media monitoring, the approach is evaluated on selected famous personalities and entities, including sports and brands.

The paper *Sentic API: A Common-Sense Based API for Concept-Level Sentiment Analysis*, by Cambria *et al.* presents *Sentic API*, which makes use of a "bag-of-concepts" model, based on ontologies and semantic networks, and a "common-sense" knowledge base, with a combination of techniques: CF-IOF, the Affective-Space vector space and the "Hourglass of Emotions", to improve on automatic extraction of semantics, sentics and sentiment from text. The authors conclude with a description of the application of Sentic API for opinion mining and sentiment analysis of patient opinion about the UK National Health Service, captured using a microblog-type feedback service.

The poster paper *Sentiment Analysis of Wimbledon Tweets*, by Sinha *et al.,* introduces novel ideas that may inspire future work on sentiment analysis on Twitter. The poster focuses on televised events where parallel annotation of video content and Twitter streams may give novel insight into the understanding of the emotional content of the events.

Micropost Classification and Extraction

Evaluating Multi-label Classification of Incident-related Tweets by A. Schulz *et al.* addresses the problem of assigning multiple labels to tweets, where such tweets are related to incidents that have occurred. The approach uses dependencies between labels to boost the performance of a multi-label classifier trained on specific label sequences. Schulz *et al.* demonstrate a good level of performance using this approach, tested on identification and classification of data concerning incidents and emergencies, with an exact-match percentage of 84.35%.

In *Combining Named Entity Recognition Methods for Concept Extraction in Microposts*, Dlugolinsky *et al.* present an approach for combining multiple named entity recognisers together. The authors demonstrate the improved performance that can be achieved, in particular in relation to recall, when using multiple, combined recognisers. We are pleased to report that Dlugolinsky *et al.* make use of the #MSM2013 Concept Extraction Challenge data[1], and reference their own and other contributions to the challenge.

Bellaachia & Al-Dhelaan, in *HG-RANK: A Hypergraph-based Keyphrase Extraction for Short Documents in Dynamic Genre*, propose an approach for extracting keyphrases from Microposts, by modeling the information as a hypergraph. The authors use a random walk approach to rank key phrases, and using the Opinosis dataset, containing Micropost-length product reviews, demonstrate the superiority of their approach with regard to state of the art baselines.

Named Entity Extraction & Linking (NEEL) Challenge

The workshop has over the years highlighted novel research directions while improving the analysis and reuse of Microposts using approaches in Information Extraction, Data Mining, Information Visualisation, Social Studies and other relevant areas. Each of these tackles these challenges from different perspectives, using a variety of state of the art and novel techniques. At the same time, the contributions to *Making Sense of Microposts* highlight the challenges still faced in research and applications using Micropost data. To respond to this challenge, #Microposts2014 hosted a *Named Entity Extraction & Linking (NEEL) Challenge*. While this and the first (*Context Extraction*) challenge, in #MSM2013, directly targeted only a sub-set of the Microposts and Social Web community, the dataset in each may be reused for other purposes, beyond information extraction and data mining. We aim to extend the challenge in the future to widen inclusion.

The #Microposts2014 NEEL challenge attracted good interest from the community, with 43 intents to submit, out of which 24 applied for a copy of the dataset, and 8 completed submission. Of these 4 were accepted, and a further 2 as posters. All challenge submissions also took part in the workshop's poster session, whose aim is to exhibit practical application in the field, and foster further discussion about the ways in which knowledge content is extracted from Microposts and reused.

The NEEL challenge was chaired by A. Elizabeth Cano and Giuseppe Rizzo, with Andrea Varga as dataset chair. Many thanks to those who helped with the annotation of the training dataset – we name these contributors in the challenge summary paper.

[1]The proceedings of the 2013 'Making Sense of Microposts' (#MSM2013) Concept Extraction Challenge are available at: `http://ceur-ws.org/Vol-1019`

We provide a brief introduction to the challenge submissions here, and more detail about the evaluation process in the challenge summary paper included in the proceedings.

Chang *et al.*, who submitted the run with the highest F_1 score, in *E2E: An End-to-end Entity Linking System for Short and Noisy Text*, present a novel approach to the NEEL task. They jointly optimised the recognition and disambiguation tasks. Based on the local and global contexts of an entity mention they generated a set of surface candidates using normalised entity lexicons, and applied overlap resolution techniques to recognise and disambiguate entity mentions.

Habib *et al.*, in *Named Entity Extraction and Linking Challenge: University of Twente at #Microposts2014*, present a sequential approach to the NEEL task by first extracting entities and then disambiguating them into a DBpedia link. They make use of state of the art features for identifying named entity (NE) candidates, including the use of Tweet segments and regular expressions. Habib *et al.* followed an NE dictionary approach for matching for the named entity linking step.

In the submission *DataTXT at #Microposts2014 Challenge*, Scaiella *et al.* propose an approach which builds on their TAGME system. They train their disambiguation algorithm with the NEEL challenge dataset. The approach in Scaiella *et al.* relies strongly on Wikipedia features for extraction and disambiguation. Their final entity linking step integrates Wikipedia categories and DBpedia RDF types as features for deploying a C4.5 classifier. *DataTXT* assigns a confidence score to each entity annotation, and discards those that fall below a specified threshold.

Yosef *et al.*, in the submission *Adapting AIDA*, extend an existing tool for entity disambiguation, AIDA. AIDA extracts entity mentions from natural language text and maps these mentions to canonical entities appearing in YAGO. To cater for Micropost content they normalise abbreviations appearing as entity mentions and supporting entity mentions appearing as username and/or hashtags. Yosef *et al.* employ a graph-based approach for linking entities, which uses different similarity measures for weighting mention-entity edges.

Bansal *et al.*, in *Linking Entities in #Microposts*, present a sequential approach to NEEL (comprising NEE + NEL). They make use of existing off-the-shelf-tools for Named Entity Extraction (NEE), and also introduce novel features for entity linking. The latter rely on the recent popularity of an entity mention on Twitter. Along with other state-of-the-art features based on Wikipedia, they applied a LambdaMART approach for the final entity disambiguation and linking step.

Finally, in *Part-of-Speech is (almost) enough: SAP Research & Innovation at the #Microposts2014 NEEL Challenge*, Dahlmeier *et al.*, present a sequential approach which makes use of off-the-shelf-tools for both NEE and NEL. They extended these toolkits using gazetteers, and employed a series of heuristics for improving the disambiguation and linking steps.

Workshop Awards

Yorkshire Tea[2], manufactured by Taylor's of Harrogate, sponsored the best paper awards. Nominations were sought from the reviewers, and a final decision agreed by the Chairs, based on the nominations and review scores. The #Microposts2014 best paper award went to:

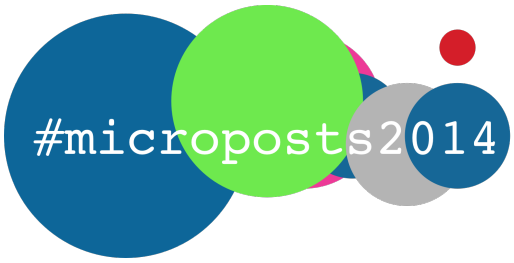
André Panisson, Laetitia Gauvin, Marco Quaggiotto & Ciro Cattuto
for their submission entitled:

Mining Concurrent Topical Activity in Microblog Streams

LinkedTV[3] sponsored the NEEL Challenge award, an iPad, for the best submission. The challenge award was also determined by the results of the quantitative evaluation. The #Microposts NEEL Challenge award went to:

Ming-Wei Chang, Bo-June Hsu, Hao Ma, Ricky Loynd & Kuansan Wang
for their submission entitled:

E2E: An End-to-end Entity Linking System for Short and Noisy Text



[2] http://www.yorkshiretea.co.uk
[3] http://www.linkedtv.eu

Additional Material

The call for participation and all paper, poster and challenge abstracts are available on the #Microposts2014 website[4]. The full proceedings are also available on the CEUR-WS server, as Vol-1141[5]. The gold standard for the NEEL Challenge is available for download[6].

The proceedings for the #MSM2013 main track are available as part of the WWW'13 Proceedings Companion[7]. The #MSM2013 Concept Extraction Challenge proceedings are published as a separate volume as CEUR Vol-1019[8], and the gold standard is available for download[9]. The proceedings for #MSM2012 and #MSM2011 are available as CEUR Vol-838[10]. and CEUR Vol-718[11], respectively.

[4] http://www.scc.lancs.ac.uk/microposts2014
[5] #Microposts2014 Proc.: http://ceur-ws.org/Vol-1141
[6] http://ceur-ws.org/Vol-1141/microposts2014-neel_challenge_gs.zip
[7] WWW'13 Companion: http://dl.acm.org/citation.cfm?id=2487788
[8] #MSM2013 CE Challenge Proc.: http://ceur-ws.org/Vol-1019
[9] http://ceur-ws.org/Vol-1019/msm2013-ce_challenge_gs.zip
[10] #MSM2012 Proc.: http://ceur-ws.org/Vol-838
[11] #MSM2011 Proc.: http://ceur-ws.org/Vol-718

Main Track Programme Committee

Gholam R. Amin Sultan Qaboos University, Oman
Pierpaolo Basile University of Bari, Italy
Julie Birkholz Vrije University, The Netherlands
Uldis Bojars SIOC Project, Latvia
John Breslin NUIG, Ireland
A. Elizabeth Cano KMi, The Open University, UK
Marco A. Casanova Pontifícia Universidade Católica do Rio de Janeiro, Brazil
Óscar Corcho Universidad Politécnica de Madrid, Spain
Ali Emrouznejad Aston Business School, UK
Guillaume Erétéo Orange Labs, France
Miriam Fernandez KMi, The Open University, UK
Fabien Gandon INRIA, Sophia-Antipolis, France
Andrés Garcia-Silva Universidad Politécnica de Madrid, Spain
Anna Lisa Gentile The University of Sheffield, UK
Robert Jäschke University of Kassel, Germany
Jelena Jovanovic University of Belgrade, Serbia
Mathieu Lacage Alcméon, France
Vita Lanfranchi The University of Sheffield, UK
Philipe Laublet Université Paris-Sorbonne, France
João Magalhães Universidade Nova de Lisboa, Portugal
Diana Maynard The University of Sheffield, UK
José M. Morales del Castillo El Colegio de México, Mexico
Fabrizio Orlandi DERI, Ireland
Alexandre Passant seevl.net / MDG Web, Ireland
Bernardo Pereira Nunes Pontifícia Universidade Católica do Rio de Janeiro, Brazil
Danica Radovanovic University of Belgrade, Serbia
Yves Raimond BBC, UK
Giuseppe Rizzo Università di Torino, Italy
Harald Sack University of Potsdam, Germany
Bernhard Schandl Gnowsis.com, Austria
Sean W. M. Siqueira Universidade Federal do Estado do Rio de Janeiro, Brazil
Victoria Uren Aston Business School, UK
Andrea Varga The University of Sheffield, UK
Shenghui Wang Vrije University, The Netherlands
Katrin Weller GESIS Leibniz Institute for the Social Sciences, Germany
Ziqi Zhang The University of Sheffield, UK

Sub Reviewers

Pavan Kapanipathi Knoesis Center, Wright State University, USA
Flavio Martins Universidade Nova de Lisboa, Portugal
Filipa Peleja Universidade Nova de Lisboa, Portugal
Víctor Rodríguez Doncel Universidad Politécnica de Madrid, Spain
Nadine Steinmetz Hasso-Plattner Institute, Germany

NEEL Challenge Evaluation Committee

Ebrahim Bagheri Ryerson University, Canada
Pierpaolo Basile University of Bari, Italy
Óscar Corcho Universidad Politécnica de Madrid, Spain
Leon Derczynski The University of Sheffield, UK
Guillaume Erétéo Orange Labs, France
Miriam Fernandez KMi, The Open University, UK
Andrés Garcia-Silva Universidad Politécnica de Madrid, Spain
Anna Lisa Gentile The University of Sheffield, UK
Robert Jäschke University of Kassel, Germany
Diana Maynard The University of Sheffield, UK
José M. Morales del Castillo El Colegio de México, Mexico
Georgios Paltoglou University of Wolverhampton, UK
Bernardo Pereira Nunes Pontifícia Universidade Católica do Rio de Janeiro, Brazil
Daniel Preoţiuc-Pietro The University of Sheffield, UK
Irina Temnikova Bulgarian Academy of Sciences, Bulgaria
Raphaël Troncy Eurecom, France
Victoria Uren Aston Business School, UK

Table of Contents

Preface i

Keynote Abstract

Computational Social Science and Microblogs – The Good, the Bad and the Ugly
Markus Strohmaier 1

Section I: Micropost Mining and Analysis

Mining Concurrent Topical Activity in Microblog Streams
André Panisson, Laetitia Gauvin, Marco Quaggiotto & Ciro Cattuto 3

TEA: Episode Analytics on Short Messages
Prapula G, Soujanya Lanka & Kamalakar Karlapalem 11

Sentic API: A Common-Sense Based API for Concept-Level Sentiment Analysis
Erik Cambria, Soujanya Poria, Alexander Gelbukh & Kenneth Kwok 19

Section II: Micropost Classification and Extraction

Evaluating Multi-label Classification of Incident-related Tweets
Axel Schulz, Eneldo Loza Mencía, Thanh Tung Dang & Benedikt Schmidt 26

Combining Named Entity Recognition Methods for Concept Extraction in Microposts
Štefan Dlugolinský, Peter Krammer, Michal Laclavík & Ladislav Hluchý 34

HG-Rank: A Hypergraph-based Keyphrase Extraction for Short Documents in Dynamic Genre
Abdelghani Bellaachia & Mohammed Al-Dhelaan 42

Section III: Poster Abstracts

Sentiment Analysis of Wimbledon Tweets
Priyanka Sinha, Anirban Dutta Choudhury & Amit Kumar Agrawal 51

Section IV: Named Entity Extraction & Linking (NEEL) Challenge

Making Sense of Microposts (#Microposts2014) Named Entity Extraction & Linking Challenge
Amparo Elizabeth Cano Basave, Giuseppe Rizzo, Andrea Varga, Matthew Rowe, Milan Stankovic & Aba-Sah Dadzie 54

Section IVa: NEEL Challenge Submissions I

E2E: An End-to-end Entity Linking System for Short and Noisy Text
Ming-Wei Chang, Bo-June Hsu, Hao Ma, Ricky Loynd & Kuansan Wang 62

Named Entity Extraction and Linking Challenge: University of Twente at #Microposts2014
Mena B. Habib, Maurice van Keule & Zhemin Zhu 64

DataTXT at #Microposts2014 Challenge
Ugo Scaiella, Michele Barbera, Stefano Parmesan, Gaetano Prestia, Emilio Del Tessandoro & Mario Verì 66

Adapting AIDA for Tweets
Mohamed Amir Yosef, Johannes Hoffart, Yusra Ibrahim, Artem Boldyrev & Gerhard Weikum 68

Section IVb: NEEL Challenge Submissions II – Posters

Linking Entities in #Microposts
Romil Bansal, Sandeep Panem, Priya Radhakrishnan, Manish Gupta & Vasudeva Varma 71

Part-of-Speech is (almost) enough: SAP Research & Innovation at the #Microposts2014 NEEL Challenge
Daniel Dahlmeier, Naveen Nandan & Wang Ting 73

Computational Social Science and Microblogs

The good, the bad and the ugly

Markus Strohmaier
GESIS & University of Koblenz
Unter Sachsenhausen 6-8
50667 Cologne, Germany
markus.strohmaier@gesis.org

ABSTRACT

According to the Computational Social Science Society of the Americas (CSSSA), computational social science is "*The science that investigates social phenomena through the medium of computing and related advanced information processing technologies*". Positioned between the computer and social sciences, this new and emerging interdisciplinary field is fuelled by at least the following two developments: (i) *availability of data*: With the web, a huge volume of social data is now available which enables the study of traces of social interactions on new scales. (ii) *increasing quantification of social theories*: With recent advances in the social sciences, social theories become increasingly formal and/or mathematical and thus amenable to quantification. Taken together, these two developments give rise to a whole range of new and interesting problems on the intersection between computer and social sciences. While a multitude of social data is available on the World Wide Web, microblogs are of particular interest due to their real-time nature, their rich social fabric and their presumed on/offline coupling. In this talk, I am going to talk about the potentials and the challenges of doing computational social science based on data obtained from microblogs such as Twitter. In particular, I want to present previous work by my group and others to identify research avenues where progress has already been made or where progress is on the horizon, and contrast these with what I feel are open research challenges in this emerging field. Work that demonstrates the potential of microblogs for computational social science includes for example [1], where we have operationalized a number of theoretical constructs from sociology to characterize the nature of online conversational practices of political parties on Twitter. In another work, we have studied the ways in which users' fields of expertise can be inferred from microblog data [4]. Work that demonstrates the pitfalls and challenges of doing computational social science with microblog data include for example [5] where we have studied a network of bots who are competing against each other in attacking users on Twitter. In subsequent work, we have found that such attacks have the potential to impact the social graph of Twitter [3], i.e. the network of who follows whom respectively who replies to whom. In other work, [2] have shown that there is a stark difference between the demographics of Twitter and the general population of the US, finding that Twitter users significantly over-represent densely populated regions and are predominantly male. I will argue that these and other factors need to be considered when we aim to unlock the full potential of microblog data for computational social science purposes.

Published as part of the #Microposts2014 Workshop proceedings, available online as CEUR Vol-1141 (http://ceur-ws.org/Vol-1141)

#Microposts2014, April 7th, 2014, Seoul, Korea.

Categories and Subject Descriptors

J.4 [**Social and behavioral sciences**]: Sociology; I.6.0 [**Simulation and Modeling**]: General

General Terms

Experimentation, Human Factors

Keywords

Social data, computational social science, social behavior, web science, online social networks

1. REFERENCES

[1] H. Lietz, C. Wagner, A. Bleier, and M. Strohmaier. When politicians talk: Assessing online conversational practices of political parties on twitter. In *International AAAI Conference on Weblogs and Social Media (ICWSM2014), Ann Arbor, MI, USA, June 2-4*, 2014.

[2] A. Mislove, S. Lehmann, Y.-Y. Ahn, J.-P. Onnela, and J. N. Rosenquist. Understanding the demographics of twitter users. In *International AAAI Conference on Weblogs and Social Media (ICWSM2011), Barcelona, Spain, July 17-21*, 2011.

[3] S. Mitter, C. Wagner, and M. Strohmaier. Understanding the impact of socialbot attacks in online social networks. In *ACM Web Science 2013, May 2-4th, Paris, France*, 2013. (Extended Abstract).

[4] C. Wagner, V. Liao, P. Pirolli, L. Nelson, and M. Strohmaier. It's not in their tweets: Modeling topical expertise of twitter users. In *The 4th IEEE International Conference on Social Computing (SocialCom 2012), Amsterdam, The Netherlands*, 2012.

[5] C. Wagner, S. Mitter, C. Koerner, and M. Strohmaier. When social bots attack: Modeling susceptibility of users in online social networks. In *Proceedings of the WWW'12 Workshop on 'Making Sense of Microposts' (MSM2012)*. CEUR-WS, 2012.

Section I:

Micropost Mining and Analysis

Mining Concurrent Topical Activity in Microblog Streams

A. Panisson, L. Gauvin, M. Quaggiotto, C. Cattuto
Data Science Laboratory, ISI Foundation, Torino, Italy
{andre.panisson},{laetitia.gauvin},{marco.quaggiotto},{ciro.cattuto}@isi.it

ABSTRACT

Streams of user-generated content in social media exhibit patterns of collective attention across diverse topics, with temporal structures determined both by exogenous factors and endogenous factors. Teasing apart different topics and resolving their individual, concurrent, activity timelines is a key challenge in extracting knowledge from microblog streams. Facing this challenge requires the use of methods that expose latent signals by using term correlations across posts and over time. Here we focus on content posted to Twitter during the London 2012 Olympics, for which a detailed schedule of events is independently available and can be used for reference. We mine the temporal structure of topical activity by using two methods based on non-negative matrix factorization. We show that for events in the Olympics schedule that can be semantically matched to Twitter topics, the extracted Twitter activity timeline closely matches the known timeline from the schedule. Our results show that, given appropriate techniques to detect latent signals, Twitter can be used as a social sensor to extract topical-temporal information on real-world events at high temporal resolution.

Keywords

topic detection, microblogs, matrix and tensor factorization, collective attention, event detection

1. INTRODUCTION

Streams of user-generated content from social media and microblogging systems exhibit patterns of collective attention across diverse topics, with temporal structures determined both by exogenous factors, such as driving from mass media, and endogenous factors such as viral propagation. Because of the openness of social media, of the complexity of their interactions with other social and information systems, and of the aggregation that typically leads to the observable stream of posts, several concurrent signals are usually simultaneously present in the post stream, corresponding to the activity of different user communities in the context of several different topics. Making sense of this information stream is an inverse problem that requires moving beyond simple frequency counts, towards the capability of teasing apart latent signals that involve complex correlations between users, topics and time intervals.

The motivation for the present study is twofold. On the one hand, we want to devise techniques that can reliably solve the inverse problem of extracting latent signals of attention to specific topics based on a stream of posts from a micro-blogging system. That is, we aim at extracting the time-varying topical structure of a microblog stream such as Twitter. On the other hand, we want to deploy these techniques in a context where temporal and semantic metadata about external events driving Twitter are available, so that the relation between exogenous driving and time-varying topical responses can be elucidated. We do not regard this as a validation of the methods we use, because the relation between the external drivers and the response of a social system is known to be complex, with memory effects, topical selectivity, and different degrees of endogenous social amplification. Rather, we regard the comparison between the time-resolved topical structure of a microblog stream and an independently available event schedule as an important step for understanding to what extent Twitter can be used as a social sensor to extract high-resolution information on concurrent events happening in the real world.

Here we focus on content collected by the Emoto project[1] from Twitter during the London 2012 Olympics, for which a daily schedule of the starting time and duration of sport events and social events is available and can be used for reference. In this context, resolving topical activity over time requires to go beyond the analysis and characterization of popularity spikes. A given topic driven by external events usually displays an extended temporal structure at the hourly scale, with multiple activity spikes or alternating periods of high and low activity. We aim at extracting signals that consists of an association of (i) a weighted set of terms defining the topic, (ii) a set of tweets that are associated to the topic, together with the corresponding users, and (iii) an activity profile for the topic over time, which may comprise disjoint time intervals of nonzero activity. We detect time-varying topics by using two independent methods, both based on non-negative matrix or tensor factorization. In the first case we build the full tweet-term-time tensor and use non-negative tensor factorization to extract the topics and their activity over time. We introduce an adapted factorization technique that can naturally deal with the special tensor structure arising from microblog streams. In the second case, which in principle affords on-line computation, we build tweet-term frequency matrices over consecutive time intervals of fixed duration. We apply non-negative matrix factorization to extract topics for each time interval and we track similar topics over time by means of agglomerative hierarchi-

[1] http://www.emoto2012.org

Published as part of the #Microposts2014 Workshop proceedings, available online as CEUR Vol-1141 (http://ceur-ws.org/Vol-1141)

#Microposts2014, April 7th, 2014, Seoul, Korea.

cal clustering.

We then apply both methods to the Twitter dataset collected during the Olympics, which reflects the attention users pay to tens of different concurrent events over the course of every day. We focus on topical dynamics at the hourly scale, and find that for those sport events in the schedule that can be semantically matched to the topics we obtain from Twitter, the activity timeline of the detected topic in Twitter closely matches the event timeline from the schedule.

This paper is structured as follows: Section 2 reviews the literature on collective attention, popularity, and topic detection in microblog streams. Section 3 describes the Olympics 2012 Twitter dataset used for the study, the event schedule we use as an external reference, and introduces some notations and conventions used throughout the paper. Section 4 and Section 5 describe the two techniques we use to mine time-varying topical activity in the Twitter stream. Section 6 discusses the relation between the time-varying topics we obtain and the known schedule of the Olympics events for one representative day, and provides some general observations on the behavior of the two methods. Finally, Section 7 summarizes our findings and points to directions for further research.

2. RELATED WORK

The dynamics of collective attention and popularity in social media has been the object of extensive investigation in the literature. Attention can suddenly concentrate on a Web page [31, 22], a YouTube video [7, 8, 20], a story in the news media [17], or a topic in Twitter [14, 2, 34]. Intrinsic features of the popular item under consideration have been related to its popularity profile by means of semantic analysis and natural language processing of user-generated content [1, 32, 33]. In particular, a great deal of research [7, 14, 15, 34, 16] has focused on characterizing the shape of peaks in popularity time series and in relating their properties to the popular item under consideration, to the relevant semantics, or to the process driving popularity.

Within the broad context of social media, Twitter has emerged as a paradigmatic system for the vision of a "social sensor" that can be used to measure diverse societal processes and responses at scale [11, 25, 3, 19]. To date, comparatively little work has been devoted to extracting signals that expose complex correlations between topics and temporal behaviors in micro-blogging systems. Given the many factors driving Twitter, and their highly concurrent nature, exposing such a topical-temporal structure may provide important insights in using Twitter as a sensor when the social signals of interest cannot be pinpointed by simply using known terms or hashtags to select the relevant content, or when the topical structure itself, and its temporal evolution, needs to be learned from the data. Saha and Sindhwani [24] adopt such as viewpoint and propose an algorithm based on non-negative matrix factorization that captures and tracks topics over time, but is evaluated at the daily temporal scale only, against events that mainly consist of single popularity peaks, without concurrency. Here we aim at capturing multiple concurrent topics and their temporal evolution at the scale of hours, in order to be able to compare the extracted signals with a known schedule for several concurrent events taking place during one day.

As we will discuss in detail, microblog activity can be represented using a tweet-term-time three-way tensor, and tensor factorization techniques can be used to uncover latent structures that represent time-varying topics. Ref. [5] proposed in 1970 the Canonical Decomposition (CANDECOM), also called parallel factorization (PARAFAC, [9]), which can be regarded as a generalization to tensors of singular value decomposition (SVD). Maintaining the interpretability of the factors usually requires to achieve factorization under non-negativity constraints, leading to techniques such as non-negative matrix or tensor factorization (NMF and NTF). Tensor factorization to detect latent structures has been extensively used in several domains such as signal processing, psychometrics, brain science, linguistics and chemometrics [26, 6, 29, 28, 30].

3. DATA AND REPRESENTATION

Notation

The following notations are used throughout the paper. Scalars are denoted by lowercase letters, e.g., x, and vectors are denoted by boldface lowercase letters, e.g., $\mathbf{x}$, where the i-th entry is x_i. Matrices are denoted by boldface capital letters, e.g., $\mathbf{X}$, where the i-th column of matrix $\mathbf{X}$ is $\mathbf{x}_i$, and the (i, j)-th entry is x_{ij}. Third order tensors are denoted by calligraphic letters, e.g., $\mathcal{A}$. The i-th slice of $\mathcal{A}$, denoted by $\mathbf{A}_i$, is formed by setting the last mode of the third order tensor to i. The (i, j)-th vector of $\mathcal{A}$, denoted by $\mathbf{a}_{ij}$, is formed by setting the second to last and last modes of $\mathcal{A}$ to i and j respectively, and the (i, j, k)-th entry of $\mathcal{A}$ is a_{ijk}.

Twitter Dataset

The Emoto dataset consists of around 14 million tweets collected during the London 2012 Summer Olympics using the public Twitter Streaming API. All tweets have at least one of 400 keywords, including common words used in the Olympic Games – like athlete, olympic, sports names and twitter accounts of high followed athletes and media companies. Tweets were collected during all the interval of 17 days comprising the Olympic Games, from July 27 to August 12 2012.

Event Schedule

In order to investigate the relation between the extracted time- varying topics and the sport events of the Olympic Games, we use the schedule available on the official London 2012 Olympics page[2], where the starting time and duration of most events is reported together with metadata about the type of event (discipline, involved teams or countries, etc.)

Data Preprocessing

For the text analysis performed in this paper, URLs are removed from the original tweet content. The remaining text is used to build a vocabulary composed of the most common 30,000 terms, where each term can be a single word, a digram or a trigram. 352 common words of the English language are also removed from the vocabulary.

In order to localize Twitter users, we examine the user profile descriptions and use an adapted version of GeoDict[3] to identify, if possible, the user country. To study the relation between the extracted topical activity and the schedule of the Olympic events, we focus on tweets posted by users located in the UK, only. This allows us to avoid potential confusion arising from tweets posted in countries, such as the USA, where Olympics events were broadcasted with delays of several hours due to time zone differences. This selection leaves us with a still substantial amount of data (about

[2]http://www.london2012.com/schedule-and-results/
[3]https://github.com/petewarden/geodict

one third of the full dataset) and simplifies the subsequent temporal analysis, even though it probably oversamples the attention payed to events that involved UK athletes.

For the scope of this study, we represent the data as a sparse third-order tensor $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$, with I tweets, J terms and K time intervals. We aggregate the tweets over 1-hour intervals, for a total of $K = 408$ intervals. The tensor $\mathcal{T}$ is sparse: the average number of terms (also referred as features in the following) for each tweet is typically no more than 10, compared to the 30k terms of our term vocabulary. Moreover, as each tweet is emitted at a given time, each interval k has a limited number of active tweets, I_k. A tensor slice $\mathbf{T}_k \in \mathbb{R}^{I \times J}$ is a sparse matrix with non-zero values only for I_k rows. $\mathbf{T}_k$ represent the sparse tweet-term matrix observed at time k. The term values t_{ijk} for each tweet i are normalized using the standard Term Frequency and Inverse-Document Frequency (TF-IDF) weighting, $t_{ijk} = \mathrm{tf}(i,j) \times \mathrm{idf}(j)$, where $\mathrm{tf}(i,j)$ is the frequency of term j in tweet i, and $\mathrm{idf}(j) = \log \frac{|D|}{1+|\{d:j \in d\}|}$ where $|D|$ is the total number of tweets and $|\{d : j \in d\}|$ is the number of tweets where the term j appears.

Visualizing Topics over Time

The methods that we present in this paper are able to extract topical-temporal structures from $\mathcal{T}$. Such topical-temporal structures can be represented a stream matrix $\mathbf{S} \in \mathbb{R}^{R \times K}$ with R topics and K intervals. Each component R is also characterized by a term-vector $\mathbf{h} \in \mathbb{R}^{J}$ that defines the most representative terms for that component. In order to visualize such topical-temporal structures represented as a stream matrix, we use the method described by Byron and Wattenberg [4], which yields a layered stream-graph visualization.

4. MASKED NON-NEGATIVE TENSOR FACTORIZATION

Problem Statement

As explained in section 3, the tensor $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$ with I tweets, J terms and K intervals is a natural way to represent the tweets and their contents with respect to the time. The tensor has the advantage to directly encompass the relationship between tweets posted at different hours and consequently between topics of the different hours. The tensor factorization as described below allows to uncover topics together with their temporal pattern.

Before describing the process of factorization itself and its output, one needs to introduce the concept of canonical decomposition (CP). CP in 3 dimensions aims at writing a tensor $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$ in a factorized way that is the sum of the outer product of three vectors:

$$\mathcal{T} = \sum_{r=1}^{R_\mathcal{T}} \mathbf{a}_r \circ \mathbf{b_r} \circ \mathbf{c_r} \tag{1}$$

where the smallest value of $R_\mathcal{T}$ for which this relation exists, is the rank of the tensor $\mathcal{T}$. In other words, the tensor $\mathcal{T}$ is expressed with a sum of rank-1 tensors. The set of vectors $a_{\{1,2,\ldots,R\}}$ (resp. $b_{\{1,2,\ldots,R_\mathcal{T}\}}$,$c_{\{1,2,\ldots,R_\mathcal{T}\}}$) can be re-written as a matrix $\mathbf{A} \in \mathbb{R}^{I \times R_\mathcal{T}}$ (resp $\mathbf{B} \in \mathbb{R}^{J \times R_\mathcal{T}}$,$\mathbf{C} \in \mathbb{R}^{K \times R_\mathcal{T}}$) where each of the $R_\mathcal{T}$ vectors is a column of the matrix. The decomposition of Eq. 1 can also be represented in terms of the three matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ as $[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$. A visual representation of such a factorization, also called Kruskal decomposition, is displayed on Fig. 1.

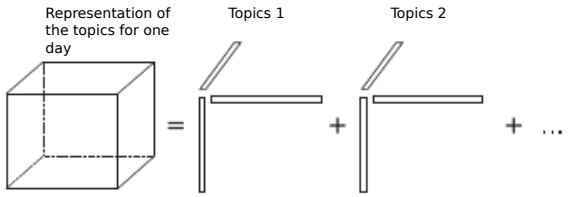


Figure 1: Representation of a Kruskal decomposition. The cube corresponds to the tensor to be factorized while the rectangles represent the vectors. In the Twitter case, each of the rank-one tensor would correspond to the description of one topics.

Factorization Methodology

Regarding the extraction of topics, the aim is not to decompose the tensor in its exact form but to approximate the tensor by a sum of rank-1 tensors with a number of terms smaller than the rank of the original tensor. This number R corresponds to the number of topics that we want to extract (see Fig. 1). Such an approximation of the tensor leads to minimize the difference between $\mathcal{T}$ and $[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$:

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \|\mathcal{T} - [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]\|_F^2 \tag{2}$$

where $\|\|$ is the Frobenius norm. We transform the 3-dimensional problem (Eq. 2) in 2-dimensional sub-problems by unfolding the tensor $\mathcal{T}$ in three different ways. This process called matricization gives rise to three modes $\mathbf{X}_{(1)}, \mathbf{X}_{(2)}, \mathbf{X}_{(3)}$. The mode-$n$ matricization consists of linearizing all the indices of the tensor except n. The three resulting matrices have respectively a size of $I \times JK$,$J \times IK$ and $K \times IJ$. Each element of the matrix $\mathbf{X}_{(i=1,2,3)}$ corresponds to one element of the tensor $\mathcal{T}$ such that each of the mode contains all the values of the tensor. Due to matricization, the factorization problem given by Eq.1 can be reframed in factorization of the three modes. In other terms, maximizing the likelihood between $\mathcal{T}$ and $[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$ is equivalent to minimizing the difference between each of the mode and their respective approximation in terms of $\mathbf{A}, \mathbf{B}, \mathbf{C}$. The factorization problem (PARAFAC) in Eq.2 is converted to the three following sub-problems where we added a condition of non-negativity of the three modes:

$$\min_{\mathbf{A} \geq 0} \|\mathbf{X}_{(1)} - \mathbf{A}(\mathbf{C} \odot \mathbf{B})^T\|_F^2 \tag{3}$$

$$\min_{\mathbf{B} \geq 0} \|\mathbf{X}_{(2)} - \mathbf{B}(\mathbf{C} \odot \mathbf{A})^T\|_F^2 \tag{4}$$

$$\min_{\mathbf{C} \geq 0} \|\mathbf{X}_{(3)} - \mathbf{C}(\mathbf{B} \odot \mathbf{A})^T\|_F^2 \tag{5}$$

where $\odot$ is the Khatri-Rao product which is a columnwise Kronecker product, i.e. such that $\mathbf{C} \odot \mathbf{B} = [c_1 \otimes b_1 c_2 \otimes b_2 \ldots c_r \otimes b_r]$. If $\mathbf{C} \in \mathbb{R}^{K \times R}$ and $\mathbf{B} \in \mathbb{R}^{J \times R}$, then the Khatri-Rao product $\mathbf{C} \odot \mathbf{B} \in \mathbb{R}^{KJ \times R}$. In our case of study, $\mathbf{A}, \mathbf{B}, \mathbf{C}$ will give each access to a different information: $\mathbf{A}$ allows to know at which topic belongs a tweet, $\mathbf{B}$ gives the definition of the topics with respect to the features and $\mathbf{C}$ gives the temporal activity of each topic.

Several algorithms have been developped to tackle the PARAFAC decomposition. The two most common are one method based on the projected gradient and the Alternating Least square method (ALS). The first one is convenient for its ease of implementation and is largely used in Singular Value Decomposition (SVD) but converges slowly. In the ALS method, the modes are deduced successively by solving Eq 5. In each iteration, for each of the sub-

problem, two modes are kept fixed while the third one is computed. This process is repeated until convergence. In our case, we use a nonnegativity constraint to make the factorization better posed and the results meaningful. One thus uses nonnegative ALS (ANLS [21]) combined with a block-coordinate-descent method in order to reach the convergence faster. Each of the step of the algorithm needs to take into account the Karush-Kuhn-Tucker (KKT) conditions to have a stationary point. Our program is based on the algorithm implemented by [12].

Masked Adaptation of the NTF

We cannot directly perform the NTF on the tensor [Tweets $\times$ Features $\times$ Interval] built as explained aboved as this tensor has a "block-disjoint" structure peculiar to the tweets. Indeed each tweet has non-zero values only at one interval because a tweet is emitted only at a given time. Each interval k has only I_k active tweets. In each slice $\mathbf{T}_k$ of the tensor, only I_k rows have meaningful values. So, we are only interested in reproducing the tensor part which contains the meaningful values. In order to focus on these meaningful values, one needs to consider an adapted version of the tensor $\mathcal{T}$. We first consider the tensor $\mathcal{T}$ built as explained above. We generate a first set of matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ which could approximate the tensor. At the next step, one tries to decompose a tensor $\bar{\mathcal{T}}$ where the values are a combination of the values of $\bar{\mathcal{T}}$ and of the values of $[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$. More exactly, this tensor has the same size than $\mathcal{T}$ and the same values than $\mathcal{T}$ for the rows I_k of each slice $\bar{\mathbf{T}}_k$. The complementary values are given by $[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$. In other terms, at each step, the tensor that we approximate is updated by:

$$\bar{\mathcal{T}} = \mathcal{T} \boxdot \mathcal{W} + (1 - \mathcal{W})[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \tag{6}$$

where $\boxdot$ is the Hadamard product (element-wise product) and $\mathcal{W}$ is a binary tensor of the same size than $\mathcal{T}$ with 1-values only when the values of $\mathcal{T}$ at this position are meaningful. The particular structure of the tensor (disjoint blocks in time) could be perceived as a "missing values" problem in the tensor, this problem has been for example tackled in [23].

Concretely, the implementation is an adaptation of a Matlab program [12] which uses the Tensor Toolbox [13]. This adaptation includes the introduction of a mask (via the tensor of weight) as mentionned above and the rewriting of some operations to avoid memory issues. This point is not detailed here as it is not part of the main point of the paper.

Stream Matrix Construction

We calculate the strength of each topic with respect to the time by using both the information about the link between each topic and each tweet and about temporal pattern of the topics. These informations are available through $\mathbf{A}$ and $\mathbf{C}$ and the consequent strength of a topic r on each interval of time k is given by:

$$s_{rk} = \sum_{i|k} a_{ir} * c_{kr} \tag{7}$$

where $\sum_{i|k}$ is a sum over the tweets indexed by i occurring at the interval indexed by k. The set of elements $s_{\{r,k\}}$ with $r = [\![1, R]\!]$ and $k = [\![1, K]\!]$ forms the stream matrix $\mathbf{S}$. Each topic is then defined by a terms vector and each of this term vector is given by a column of $\mathbf{B}$.

5. AGGLOMERATIVE NON-NEGATIVE MATRIX FACTORIZATION

Non-negative Matrix Factorization

For each tensor slice $\mathbf{T}_k$, we compute a non-negative factorization by minimizing the following error function,

$$\min_{W,H} \|\mathbf{T}_k - \mathbf{W}^{(k)}\mathbf{H}^{(k)}\|_F^2, \tag{8}$$

where $\|\|$ is the Frobenius norm, subject to the constraint that the values in $\mathbf{W}^{(k)}$ and $\mathbf{H}^{(k)}$ must be non-negative. The non-negative factorization is achieved using the projected gradient method with sparseness constraints, as described in [18, 10]. The factorization produces a matrix of left vectors $\mathbf{W}^{(k)} \in \mathbb{R}^{I_k \times F}$ and a matrix of right vectors $\mathbf{H}^{(k)} \in \mathbb{R}^{F \times J}$, where F is the number of components used in the decomposition. The matrix $\mathbf{H}^{(k)}$ stores the term vectors of the extracted components at interval t. The matrix $\mathbf{W}^{(k)}$ is used to calculate the strength of each extracted component, which are represented in a matrix $\mathbf{Z} \in \mathbb{R}^{F \times K}$ given by

$$z_{fk} = \sum_{i=1}^{I_k} \frac{w_{if}^{(k)}}{\sum_{f'=1}^{F} w_{if'}^{(k)}} \tag{9}$$

where z_{fk} is the strength of factor f at interval k.

Component Clustering

In order to track topics over time, we need to merge components into topics depending on how similar they are. Since each component is defined by a term vector, we can calculate a similarity matrix of all possible pairs of term vectors using cosine similarity. This matrix is fed to a standard agglomerative hierarchical clustering algorithm, known as UPGMA [27], that at each step combines the two most similar clusters into a higher-level cluster. Cluster similarity is defined in terms of average linkage: that is, the distance between two clusters c_1 and c_2 is defined as the average of all pair-wise distances between the children of c_1 and those of c_2.

The hierarchical clustering produces a tree that can be cut at a given depth to yield a clustering at a chosen level of detail. That is, by varying the threshold similarity we use for the cut we can go from a coarse-grained topical structure, with few clusters that may merge unrelated topics, to a fine-grained topical structure, with many clusters that may separate term vectors that otherwise could be regarded as the same continuous topic over time. The cut threshold needs to be chosen based on criteria that depends on the application at hand.

Each choice for the cut yields a number of clusters C and a map function $\mathcal{C}(r, f) \rightarrow k$ that associates the component index f at time interval k to a topic cluster r. This function collects all components associated to cluster r in a set $\mathbb{C}_{rk}$ for each interval k.

Stream Matrix Construction

When constructing the stream matrix, the number of topics R in the stream matrix is given by the number of clusters generated by the clustering step. In order to calculate the entries s_{rk} of the resulting stream matrix $\mathbf{S}$, we aggregate the strengths of the clustered components. We build a stream matrix $\mathbf{S} \in \mathbb{R}^{R \times K}$, with R topics and K intervals, given by

$$s_{rk} = \sum_{f \in \mathbb{C}_{rk}} z_{fk} \tag{10}$$

Finally, we extract the term vectors that are associated to each clus-

ter. Each cluster will be associated to a term vector $\mathbf{h}_r^{(k)} \in \mathbb{R}^J$ that is the average of all term vectors $\mathbf{h}_f^{(k)}$ associated to that cluster in the component clustering step.

6. ANALYSIS OF THE OLYMPICS DATASET

We now move to the analysis of the London 2012 Twitter dataset and its relation with the known schedule of the Games. We focus on one representative day, July 29th, during which several sport events took place at different times and concurrently. We use both topic detection methods, show the signals they extract, and check to what extent they are capable of extracting signals that we can understand in terms of the schedule.

The topic detection methods are set up as follows. For the masked NTF method, we decompose the tensor using a fixed number of components, using a tolerance value of 10^{-4} for the stopping condition, and limiting the number of iterations to 50. For the agglomerative NMF method, we decompose each interval matrix using a fixed number of components. We use a tolerance value of 10^{-4} for the stopping condition, and limit the number of iterations to 20. We use 250 topics for the Masked NTF, and 50 components per time intervals in the Agglomerative NMF.

Figure 2 shows a streamgraph representation of time-varying topics extracted using the two methods we have discussed. Two global activity peaks are visible in both streamgraphs: the peak at about 2.30pm UTC was triggered when Elizabeth Armistead won the silver medal in road cycle; the peat at about 7pm UTC is driven by the bronze medal in 400m freestyle to Rebecca Adlington. In the stream graphs, for clarity, each topic is annotated using only its topmost weighted term. This makes it difficult to assess a visual correspondence between the same topics across the two representations, as the term with top weight may be different for the two term vectors even though the vectors are overall very similar (in terms of cosine similarity). On closer inspection, many precise correspondences can be established between the topics extracted by the Masked NTF method and those extracted by the Agglomerative NMF method: for example, the topic *armistead* in the top streamgraph matches the topic *congratulation* in the bottom one. An interactive streamgraph visualization of the London 2012 Twitter dataset is available at http://www.datainterfaces.org/projects/emoto/.

6.1 Comparison with the Olympics Schedule

Event Selection

In order to show the possible correspondence between the extracted topics and sport events, we manually annotate the schedule collected from the official London 2012 Olympics page for July 29th, 2012. As the number of events in a day can be substantial and we want to focus on events with higher impact on social media, we retain events that are either finals or team sports match. We annotate each event with a set of at most three terms extracted from the schedule, as described in Section 3. For a team sport, we use the sport name and the countries of the two teams, otherwise, we put the name of the sport and its characteristics, e.g., the discipline for swimming.

Matching Topics and Events

For each event, we use a matching criteria to select one of the extracted topics from each of the set of topics produced by the methods. Since we want to select a topic in which all event annotated terms appear with a high weight in its term vectors, we define our matching score based on the geometric average of the weights of the event annotated terms in the topic's term vectors:

$$\langle w \rangle = \sqrt[n]{h_{w_1 r}\, h_{w_2 r}\, \ldots\, h_{w_n r}} \tag{11}$$

For Masked NTF, for each event, we choose the topic with the highest corresponding geometric average $\langle w \rangle$. In the agglomerative NMF case, for each event, we choose the topic with the highest corresponding geometric average $\langle w \rangle$ weighted by $\log(n)$ where n is the number of components in the selected cluster. We use $\log(n)$ in order to favor the selection of clusters with a higher number of aggregated components, otherwise the most detailed clusters which aggregates only one component are always selected. Since the Agglomerative NMF method produces a tree structure in which each node agglomerates a set of components and represents topic activity, we have to calculate such matching result for each node, and select the node for which such matching result is the highest.

Results and Observations

At this point, we have, for each event, a topic which was selected in each method, and the corresponding matching result. In Figure 3, we show the schedule events for the top 20 highest matching results. In the lefthand figure, we show, for each one of the top 20 matching results, the topic extracted by the Masked NTF method, while in the righthand figure, we show the topic extracted by the Agglomerative NMF method. The results are sorted by the corresponding matching weight.

For each event, on its top left corner, we show the manually annotated terms used for the matching. The shaded blue area shows the exact interval during which the event was occurring according to the official Olympics schedule. In the same area, the solid green line represents the temporal structure of the topic with higher matching result according to our matching criteria. Such values roughly represent the amount of activity for such topic and are normalized according to the peak of activity. We show the value for this peak in the top right side, along with the matching results between parenthesis. In the Agglomerative NMF graph (on the right) we show as a dotted line the activity in time for the given terms regarding the number of tweets that have such terms (tweet count). We remark that by considering only the dotted line the timing of many events on the right side of the figure does not match the schedule timings, i.e., merely counting tweets is not sufficient at this resolution level. We also measured the number of tweets where the terms are co-ocurring, and in this case the number of tweets is so small that it does not allow the detection of any structure in time.

We evaluated these activity profiles using the CrowdFlower Web-based crowdsourcing platform (restricted to Amazon Mechanical Turk workers). Each work unit asks a worker to visually inspect and compare two timelines: the one to be evaluated, and a reference timeline corresponding to the known time intervals for sport events taken from the Olympic schedule. Each work unit looks like a row from Figure 3. Our evaluation was based on 100 work units evenly distributed among 5 types: 1) (NMF) work units based on the results of Agglomerative NMF; 2) (CNT-NMF) work units with activity profiles generated by simply counting the number of tweets with the terms used in matching the NMF topics; 3) (NTF) work units from the Masked NTF approach; 4) (CNT-NMF) same as (CNT-NMF) for Masked NTF; 5) synthetic work units ("gold" units) used to assess worker quality. For each work unit, we asked the workers whether the two timelines matched exactly (Yes), matched partially (Partially) or not at all (No). 95% of the judgments

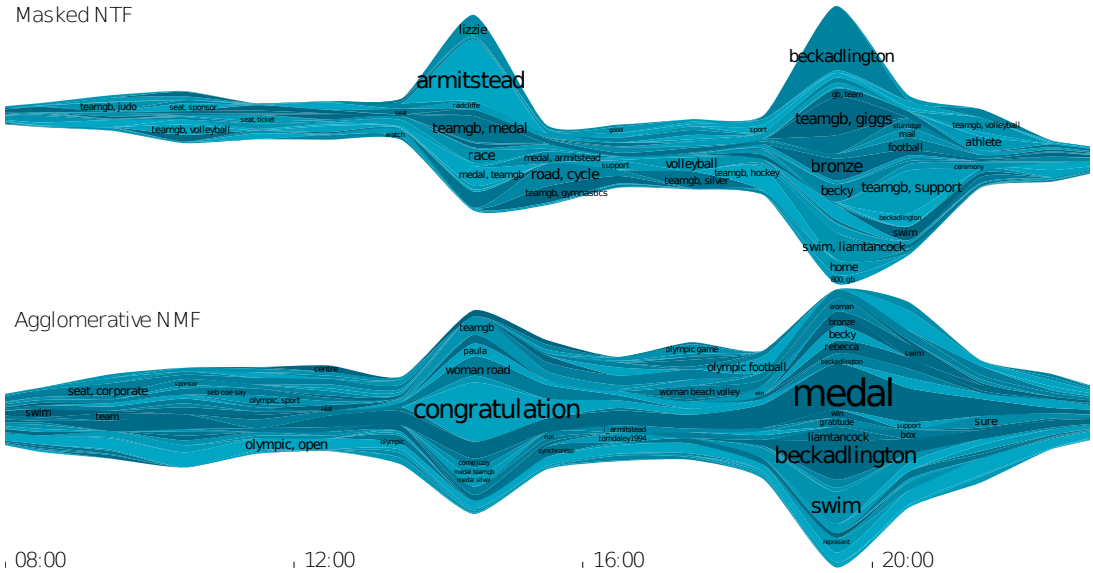


Figure 2: Streamgraph visualization of the stream matrices generated by the Masked NTF (top) using 50 topics, and by the Agglomerative NMF (bottom) using 20 components per interval and a total of 150 clusters. Interactive streamgraph visualizations for a few use cases are available at http://www.datainterfaces.org/2013/06/twitter-topic-explorer/.

for gold work units were correct. We only retained those users who correctly judged more than 80% of the gold units. Figure 4 shows the distribution of judgements for the different types of work units. The left hand side of the figure shows the distribution obtained for all work units, while the right hand side shows the distribution restricted to work units with more than 80% of agreement across different workers. According to this evaluation, both NTF and NMF outperform the count-based methods.

We see that for most of the events there is a close temporal alignment between the event schedule and the topic structure, at the scale of the hour or less. We see that such temporal alignment is much closer than when compared to the peaks of activity generated by counting tweets.

We observe that the mismatches in the temporal alignment are caused by two different factors. The first one is due to a low matching results, like the event annotated with (football, mexico, gabon). It means that the term vectors for the given topic does not represent with high confidence the terms used to annotate the event. The second one is due to a different behaviour in collective attention. This happens for example in the case of swimming events, where the first part of the event is related to eliminatories and the second part is related to the finals. In such cases, the peak in activity arrives when the event finishes and the attention goes to the winner.

7. SUMMARY AND FUTURE WORK

The topic detection techniques we discussed here afford tracking the attention that a community of users devotes to multiple concurrent topics over time, teasing apart social signals that cannot be disentangled by simply measuring frequencies of term or hashtags occurrences. This allows to capture the emergence of topics and to track their popularity with a high temporal resolution and a controllable semantic granularity. The comparison with an independently available schedule of real-world events shows that the response of Twitter to external driving retains a great deal of temporal and topical information about the event schedule, pointing to more sophisticated uses of Twitter as a social sensor.

The work described here can be extended along several directions. It would be interesting to develop and characterize on-line versions of the techniques we used here, so that topic emergence and trend detection could be carried out on live microblog streams. Because of its temporal segmentation, the Agglomerative NMF case lends itself rather well to on-line incremental computation, whereas a dynamic version of the Masked NTF technique would be more challenging to achieve.

Another interesting direction for future research would be to augment the tweet-term-time tensor with a fourth dimension representing the location of the users, so that the latent signals we extract could expose correlation between topics, time intervals and locations, exposing geographical patterns of collective attention and their relation to delays, e.g., in the seeding by mass media across different countries.

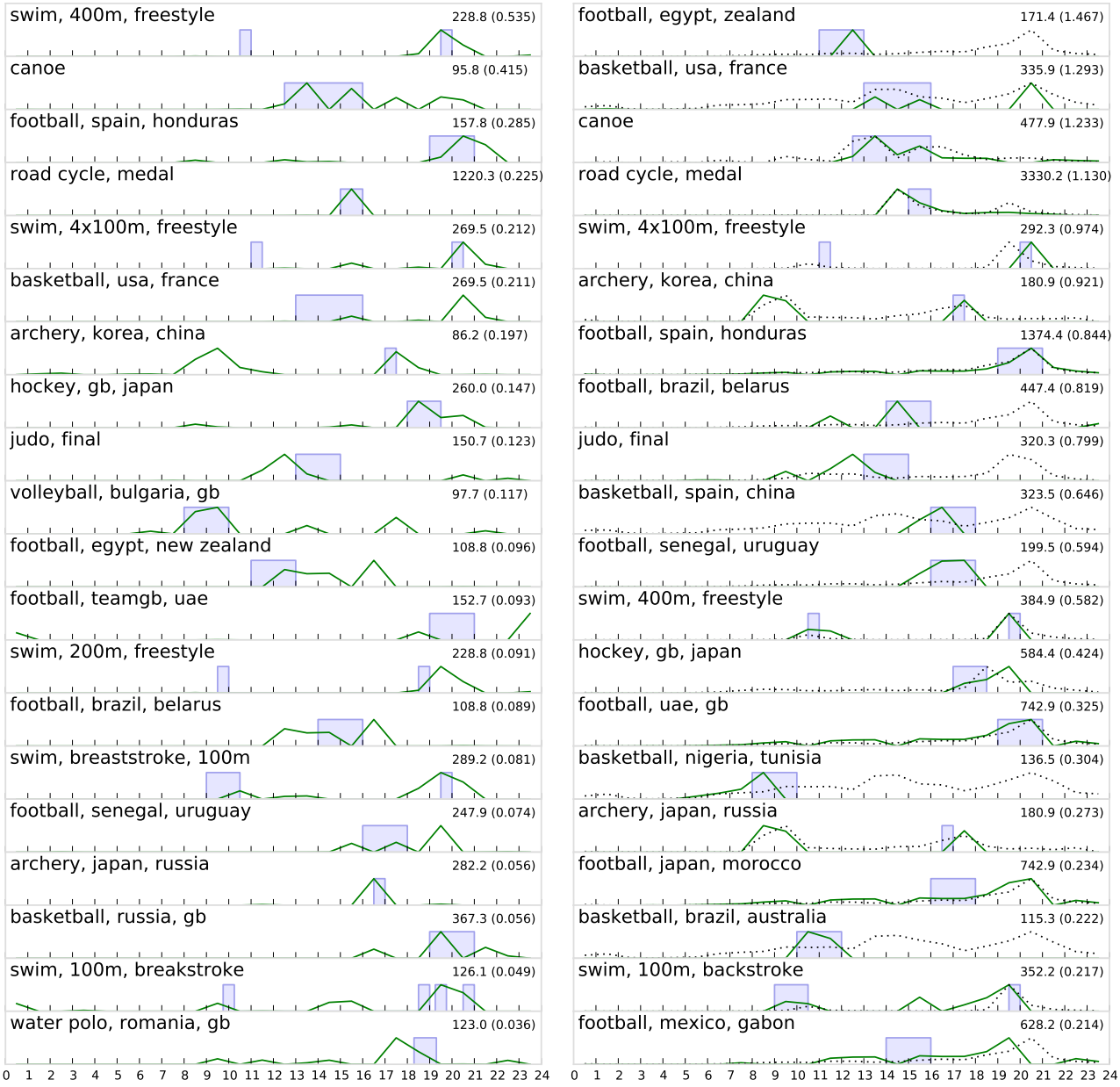
Acknowledgements

The Authors acknowledge the Emoto project www.emoto2012.org and its partners for access to the Twitter dataset on the London Olympics 2012. The Authors acknowledge inspiring discussions with Moritz Stefaner and Bruno Goncalves. The Authors aknowledge partial support from the Lagrange Project of the ISI Foundation funded by the CRT Foundation, from the Q-ARACNE project funded by the Fondazione Compagnia di San Paolo, and from the FET Multiplex Project (EU-FET-317532) funded by the European Commission.

8. REFERENCES

[1] E. Adar, D. Weld, Bershad, B.N., and S. Gribble. Why we search: visualizing and predicting user behavior. In *Proc. 16th intl. conf. on World Wide Web (WWW'07)*, pages 161–170, 2007.

[2] S. Asur, B. A. Huberman, G. Szabo, and W. C. Trends in social media : Persistence and decay. In *Proc. 5th Intl. Conf. on Weblogs and Social Media (ICWSM)*, page 434, 2011.



(a) Masked NTF

(b) Agglomerative NMF

Figure 3: The top 20 most representative schedule events regarding the matching weight of its annotations with the term vectors of an extracted topic. In the left, for each event, we show the topic extracted by the Masked NTF method for which the matching weight is the highest, and in the right we show the topic extracted by the Agglomerative NMF method for which the matching weight is the highest. Since we are showing the topmost 20 schedule events regarding the matching weight, the events are sorted by such matching weight. On the top left corner of each event, we show its annotated terms, along with the exact interval in which the event happened according to the official Olympics schedule (shaded blue area). The solid green line shows the temporal structure of the topic with higher matching weight along the 24 hours of July 29. The values in the top right side shows the value for the peak of the temporal structure, which roughly represents the amount of activity for such topic, and, between parenthesis, the matching weight for the given topic. In the Agglomerative NMF side (on the right) we show as a dotted line the activity in time for the given terms regarding the number of tweets that have such terms (tweet count).

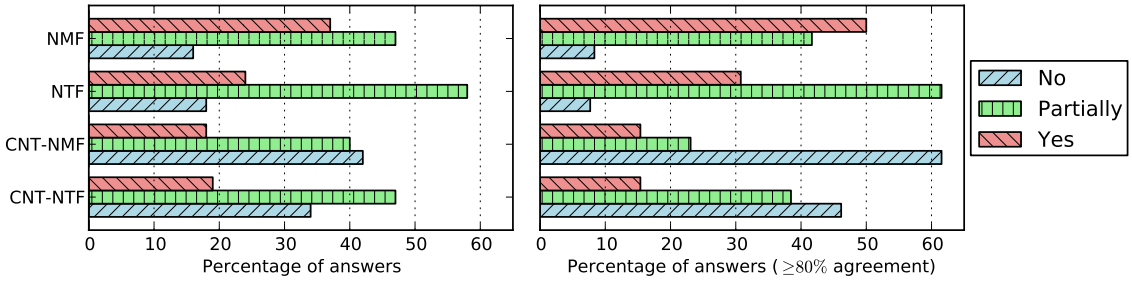


Figure 4: Crowdsourced evaluation of the topical activity profiles for selected sport events (see main text) of the London 2012 Olympics dataset obtained by using the different topic detection methods.

[3] J. Bollen, H. Mao, and X. Zeng. Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1):1 – 8, 2011.

[4] L. Byron and M. Wattenberg. Stacked graphs–geometry & aesthetics. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1245–1252, 2008.

[5] J. Carroll and J.-J. Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika*, 35(3):283–319, September 1970.

[6] A. Cichocki, A. H. Phan, and R. Zdunek. *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*. Wiley, Chichester, 2009.

[7] R. Crane and D. Sornette. Robust dynamic classes revealed by measuring the response function of a social system. *PNAS*, 105:15649, 2008.

[8] F. Figueiredo, F. Benevenuto, and J. Almeida. The tube over time: Characterizing popularity growth of youtube videos. In *Proc. ACM Intl. Conf. on Web Search and Data Mining (WSDM)*, pages 745–754, 2011.

[9] R. A. Harshman. Foundations of the PARAFAC procedure: Models and conditions for an" explanatory" multi-modal factor analysis. *UCLA Working Papers in Phonetics*, 16(1):84, 1970.

[10] P. Hoyer. Non-negative matrix factorization with sparseness constraints. *The Journal of Machine Learning Research*, 5:1457–1469, 2004.

[11] B. J. Jansen, M. Zhang, K. Sobel, and A. Chowdury. Twitter power: Tweets as electronic word of mouth. *Journal of the American Society for Information Science & Technology*, 60(11), 2009.

[12] J. Kim and H. Park. Fast nonnegative tensor factorization with an active-set-like method. In M. W. Berry, K. A. Gallivan, E. Gallopoulos, A. Grama, B. Philippe, Y. Saad, and F. Saied, editors, *High-Performance Scientific Computing*, pages 311–326. Springer London, 2012.

[13] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3):455–500, Aug. 2009.

[14] H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? *WWW '10 Proc. of the 19th intl. conf. on World wide web*, page 591, Feb 2010.

[15] D. Laniado and P. Mika. Making sense of twitter. In *Semantic Web - ISWC*, volume 6469, pages 470–485, 2010.

[16] J. Lehmann, B. Gonçalves, J. J. Ramasco, and C. Cattuto. Dynamical classes of collective attention in twitter. In *Proc. of the 21st intl. conf. on World Wide Web*, WWW '12, pages 251–260, New York, NY, USA, 2012. ACM.

[17] J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *Proc. of the 15th ACM SIGKDD intl. conf. on Knowledge discovery and data mining*, page 497, 2009.

[18] C. Lin. Projected gradient methods for nonnegative matrix factorization. *Neural computation*, 19(10):2756–2779, 2007.

[19] D. Mocanu, A. Baronchelli, N. Perra, B. Gonçalves, Q. Zhang, and A. Vespignani. The twitter of babel: Mapping world languages through microblogging platforms. *PloS one*, 8(4):e61981, 2013.

[20] M. Naaman, H. Becker, and L. Gravano. Hip and trendy: Characterizing emerging trends on twitter. *J. Am. Soc. Inf. Sci.*, 62:902–918, 2011.

[21] P. Paatero and U. Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126, 1994.

[22] J. Ratkiewicz, F. Menczer, S. Fortunato, A. Flammini, and A. Vespignani. Traffic in social media ii: Modeling bursty popularity. In *SocialCom 2010: SIN*, 2010.

[23] J.-P. Royer, N. Thirion-Moreau, and P. Comon. NonNegative 3-Way tensor Factorzation taking into account Possible Missing Data. In Eurasip, editor, *EUSIPCO-2012*, pages 1–5, Bucarest, Roumanie, Aug. 2012. Elsevier.

[24] A. Saha and V. Sindhwani. Learning evolving and emerging topics in social media: a dynamic nmf approach with temporal regularization. In *Proc. of the fifth ACM intl. conf. on Web search and data mining*, WSDM '12, pages 693–702, New York, NY, USA, 2012. ACM.

[25] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proc. of the 19th intl. conf. on World wide web*, WWW '10, pages 851–860, New York, NY, USA, 2010. ACM.

[26] A. Shashua and T. Hazan. Non-negative tensor factorization with applications to statistics and computer vision. In *Proc. of the 22nd intl. conf. on Machine learning*, ICML '05, pages 792–799, New York, NY, USA, 2005. ACM.

[27] R. R. Sokal and C. D. Michener. A statistical method for evaluating systematic relationships. *University of Kansas Scientific Bulletin*, 28:1409–1438, 1958.

[28] J. Sun, D. Tao, and C. Faloutsos. Beyond streams and graphs: dynamic tensor analysis. In *Proc. of the 12th ACM SIGKDD intl. conf. on Knowledge discovery and data mining*, KDD '06, pages 374–383, New York, NY, USA, 2006. ACM.

[29] T. Van de Cruys. A non-negative tensor factorization model for selectional preference induction. In *Proc. of the Workshop on Geometrical Models of Natural Language Semantics*, GEMS '09, pages 83–90, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[30] Y. Wang and E. Agichtein. Temporal latent semantic analysis for collaboratively generated content: preliminary results. In *Proc. of the 34th intl. ACM SIGIR conf. on Research and development in Information Retrieval*, SIGIR '11, pages 1145–1146, New York, NY, USA, 2011. ACM.

[31] F. Wu and B. A. Huberman. Novelty and collective attention. *Proc. Nat. Acad. Sci.*, 104:17599, 2007.

[32] J. Wu, K. M. Thornton, and E. N. Efthimiadis. Conversational tagging in twitter. In *Proc. of the 21st ACM conf. on Hypertext and Hypermedia*, pages 173–178, 2010.

[33] S. Wu, J. M. Hofman, W. A. Mason, and D. J. Watts. Who says what to whom on twitter. *WWW 2011*, pages 1–10, Feb 2011.

[34] J. Yang and J. Leskovec. Patterns of temporal variation in online media. In *Proc. of the fourth ACM intl. conf. on Web search and data mining*, pages 177–186, 2011.

TEA: Episode Analytics on Short Messages

Prapula G
Center for Data Engineering
IIIT Hyderabad
Andhra Pradesh, India
prapula.g@research.iiit.ac.in

Soujanya Lanka
Center for Data Engineering
IIIT Hyderabad
Andhra Pradesh, India
soujanya@iiit.ac.in

Kamalakar Karlapalem
Center for Data Engineering
IIIT Hyderabad
Andhra Pradesh, India
kamal@iiit.ac.in

ABSTRACT

Twitter is a widely used micro-blogging service, which in recent times, has become a reliable source of happening news around the world [11]. Breaking news are covered in twitter; the magnitude and volumes of tweets reflecting on the nature and intensity of the news. During events, many tweets are posted either expressing sentiments about the event or just about the occurrence of the event. Events related to an entity that have attracted a large number of tweets can be considered significant in the entity's twitter lifetime. Entity could represent a person, movie, community, electronic gadgets, software products and like wise. In this work, we attempt to automatically detect significant events related to an entity. An episode, is an event of importance; identified by processing the volumes of tweets/posts in a short time.

The key features implemented in Tweet Episode Analytics (TEA) system are: (i) detecting episodes among the streaming tweets related to a given entity over a period of time (from the entity's birth i.e., mention in the tweet world till date), (ii) providing visual analytics (like sentiment scoring and frequency of tweets over time) of each episode through graphical interpretation.

Categories and Subject Descriptors

H.4 [**Web IR and Social Media Search**]: Social Network Analysis(Micro-Blogging Analysis)

General Terms

Entity, Trend, Events, Sentiment, Analysis, Detection

Keywords

Tweets, *Episode*, Text Analytics

1. INTRODUCTION

Tweets are a source of valuable information that have the potential of providing an overview of how the world is thinking about various events/persons over a period of time. The events are usually related to nouns like persons, movies and objects in real world; these nouns are referred to as entities. Each entity will have a series (one or more) of events which are significant in its lifetime. People tweet about events that are of importance to them[16][13]. People seek latest up-to-date information by searching through tweets live stream. So, an event or a search phrase obtains a high frequency of tweets, mostly due to its significance (like a trending topic). Hence, the overall social interest received for an event related to an entity is reflected by the number of tweets that mention the event. This streaming information about various events should be identified, analyzed and visualized in order to make them suitable for humans to understand and interpret the causes and the consequences. Such a visual representation is also useful in displaying search results. AspecTiles[10] address the problem of search result diversification. In our work, *given an entity we address the event diversification related to an entity.* For instance, if a search on 'Roger Federer' is performed during the Wimbledon season, there could be various events related to Federer that would have been tweeted on different days of the season. Identifying significant events and displaying sets of tweets (by grouping tweets related to a particular event) with graphs gives user a chance to glance through events and explore in detail on an event he/she is interested in.

With large number of twitter users getting interested in a particular event leads to a deluge of tweets and also the queries on those tweets. Mining significant events will be useful in summarizing the deluge of tweets. Hence, an analysis system is needed, that (i) identifies important events related to an entity, (ii) analyzes the temporal sentiment patterns of tweets during the period of increased interest and provides visuals depicting the same. A large scale processing is done to accomplish all of this and the results of each of the above is presented in Section 5.

The importance of an event can be computed by the frequency of tweets and re-tweet counts related to the event as done in [14]. A popular entity (like a movie star, movie, musician and the likes) receives some amount of attention on a regular basis in twitter. The amount of attention received need not to be constant over a daily basis. The attention received (i.e., the number of tweets talking about the entity) varies over a period of time due to various events related to the entity. When there is a spike in the attention received, the event associated could be a significant one.

For instance, let us consider 'Lady Gaga' as an entity. There could be many tweets that mention Lady Gaga as part of routine events like '@user432 Listening to Lady Gaga',

Published as part of the #Microposts2014 Workshop proceedings, available online as CEUR Vol-1141 (http://ceur-ws.org/Vol-1141)

#Microposts2014, April 7th, 2014, Seoul, Korea.

'just read article on Lady Gaga', 'Lady Gaga in Japan' and 'Lady Gaga's Born this way - releasing in 2012'. Among these, significant events for Lady Gaga could be 'Born this way' album's release and her 'tour to Japan'. A significant event due to increased volumes of tweets related to an entity is considered as an episode.

The sentiments expressed by twitter users about episodes change over time. For example, there could be a very positive anticipation for a particular movie about to be released, but it might not have been well received (paving way for negative sentiments expressed post-release). Analyzing and visualizing the accumulated sentiments about episodes over time could be useful for market research analysis of an entity (movies, electronic gadgets, albums etc).

In this paper, we introduce the concept of an episode for a time-line of an entity and develop a tweet episode analytics system (referred to as TEA) which when given a phrase of words that represent an entity as input can: (a) identify episodes, (b) analyze episodes, life-spans, (c) display the cumulative sentiments expressed over a period of time.

In section 2, we present related work. In section 3, an Overview of TEA is presented which is followed by Tweet Episode Analytics (Section 4). Section 5 presents Results of TEA with Section 6 presenting some conclusions.

2. RELATED WORK

There has been a considerable amount of work done on extracting trending topics from twitter. The idea of an Episode that has been proposed in this paper is different from the past studies on trending topics. There has been a study on how and why the topics become trending in one of the papers [6]. As a part of their study, [6] have tried to explain the growth of trending topics. They have concluded that most topics do not trend for long on Twitter. This conclusion from their study strengthens our idea of Episodes which we have defined as a significant event that may occur in the time line of an entity and the event will be significant only for a short period of time.

In [7], Becker *et al* identified real-world events and their associated twitter messages that are published. Online clustering and filtering framework is used to address this event identification problem.We have introduced the concept of an episode and have presented an algorithm to identify an episode by considering accumulated significance of the tweets.

In [14], Nichols *et al* extracted sporting events and summarized the tweets in that events. They are confined to tweets related to sports and concentrated more on summarizing than extracting events. Our frame work and algorithm work for a search query (to represent the entity) and detect possible episodes in its life time.

In [15], Sakaki *et al* believe that when a real event like natural disasters that influence people from either one region or some parts of the world occur, the twitter users (social sensors) will tweet about the event immediately. This paper aims to recognise events at real time whereas we detect episodes that have already occurred and have lots of importance in the entity's life time. Our paper presents historical coverage of an entity as a sequence of episodes. Moreover, this paper targets events like social events (e.g., large parties), sports events, accidents and political campaigns and natural events like storms, heavy rainfall, tornadoes, typhoons which influence people's daily life whereas our work is not specific to any event of an entity and is more generic.

In [9], Gruhl *et al* studied the propogation of information in environments like personal publishing using a large collection of web logs. They have characterized the topics into long running "chatter" topics consisting of recursive "spikes" topics. According to their theory, if there are spikes recursively for a topic over a long period of time, it may be of interest. Topics are detected and then classified if its chatter or spike and studied the propagation. Our work concentrates on detecting events related to an entity based on a similar notion that spikes are the places where significant events have occurred in an entity's life time.

3. OVERVIEW OF TEA

In this section, we introduce the concept of an episode. We also present the architecture of "Tweet Episode Analytics" system as a part of this section.

3.1 What is an *Episode*?

Episode can be defined as a significant event in the time line of an entity (individual person, community, group etc) that has occurred due to a sudden increase of tweet volumes of the entity from its regular volumes.

Among all the events that an object/entity is involved in, the events that received more attention in a particular period of time, are referred as episodes. All episodes are events but not all events can be episodes. Episodes are significant events with respect to an entity, but events are more general not specifically related to entities. Episodes are always for an entity. TEA algorithm identifies prominent episodes of an entity that has occurred over its time line, considering an entity has a long lifespan. An episode is different from the traditional concept of "a trending topic" [12] or "topics extracted from topic clustering" [8]. An entity is said to have an episode if there is a sudden spike in an activity and that is captured as an event in the time line of the entity because of which there is a huge activity related to the entity. For each such event, there is evidence like an article or information that shows the true importance of the event. If no such article or information exists, then it may not be an episode.

Similar to 'Lady Gaga' example mentioned in Section 1, we noticed a similar episode being detected in our tweet data set related to 'Justin'(entity). A phrase formed by 'Justin' and 'Boyfriend' put together is an episode whereas 'Justin' is not. After the release of Justin Bieber's new song 'Boyfriend', there was a sudden outburst of tweets about this song. Even though the number of tweets about 'Justin' are large implying that it is a trending topic, it is not an episode because the reason for more social activity about 'Justin' is not due to a single significant event.

3.2 System Architecture

The whole tweet episode analytics system can be divided into different modules. Tweet collection and tweet processing are offline modules (module in which processing is done beforehand) where as, episode detection, sentiment analyzing are online modules (module in which the processing starts after receiving the query as input to the system). The flowchart of system architecture to "Detect Episodes of an entity from Twitter data using Episode Detection Algorithm" is given in Figure 1. Below is a brief explanation for each of the modules.

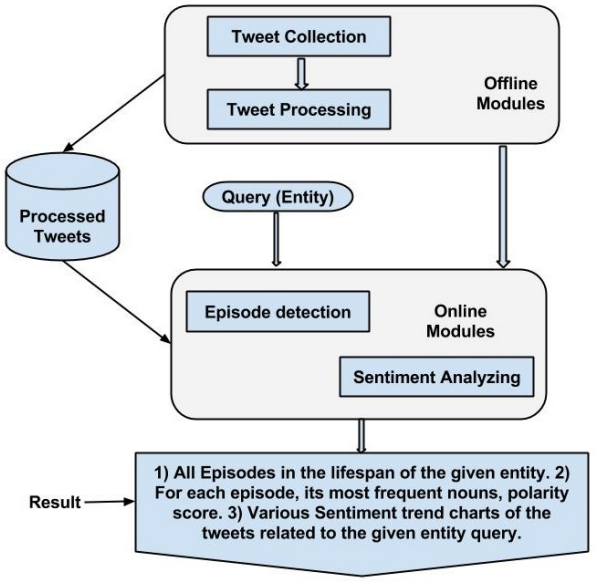


Figure 1: Flow Chart to detect episodes from Tweets using Episode Detection Algorithm

3.2.1 *Tweet Collection Module*

Tweet Collection module collects tweets using *Twitter Streaming API*. A sample of public tweets are extracted from twitter.com every 2 minutes. We have been collecting tweets since March 2012 and until December 2012. Around *140 Million* public tweets were collected from Twitter. Tweets were collected on an hourly basis; tweets for each hour are stored in a separate file.

3.2.2 *Tweet Processing Module*

Tweet processing includes removing non-english tweets and tweets with incomplete details. These processed tweets are stored by indexing them using Lucene [1]. The details about a tweet that are being stored in the Lucene index are tweet id, text, retweet count of that particular tweet and its creation time. In addition to this, the id, name, location, url, description, followers count, creation time of the account of the user who has tweeted the tweet are also stored for each tweet.

3.2.3 *Episode Detection Module*

A query(entity) is given as input to this module along with the processed Lucene Index from the above module. Episode detection module will extract all the tweets that are related to the given query and then all the episodes that have occurred over the life time of the entity are detected by applying Episode Detection Algorithm on the related tweets.

3.2.4 *Sentiment Analysing Module*

Sentiment Analysis is a method of analyzing/finding the opinion/sentiment that is expressed in a piece of text, a tweet in our context. In this module, a very basic sentiment scoring algorithm is applied on the tweets which are related to the given entity to get their sentiment score. This algorithm could be replaced with any other sentiment scoring algorithm; for this paper, we used a basic scoring algorithm as explained in Section 4.3. This module generates charts/graphs which shows how the sentiment of the entity has been changing over the period of its twitter lifetime.

We have given "Federer" query for our system along with the output of Tweet Collection and Tweet Processing offline modules and the flow is as below: (i) we retrieved episodes mentioned in Table 1 using Episode detection module, (ii) from episodes - we merged episodes and got bubble chart, (iii) we extracted sentiment scores and the trending graphs using sentiment analysis module.

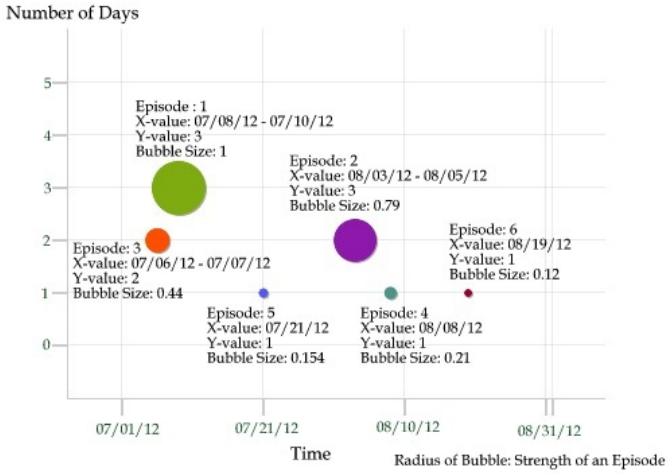


Figure 2: Episodes strength chart of entity "Federer" (see Equation 5)

4. TWEET EPISODE ANALYTICS

In this section, we present our algorithm to detect episodes from the tweet data. After the episode detection algorithm is executed on the data set, we use the information obtained from the algorithm to detect all the episodes of a particular entity. We also present sentiment analysis method that we have used in our system. In the post processing phase, we present sentiment, trend and temporal analytics of each episode.

4.1 Episode Detection Algorithm

Given an entity/query as an input, Episode Detection Algorithm gives episodes for an entity over a given time period. The algorithm will detect the episodes that have occurred in the entity's twitter lifetime. The time of birth for an entity in our twitter data set is the time stamp of the first occurring tweet that mentions it. Lifetime of an entity would be the first time stamp to till date. For this, all the tweets related to a given query are extracted from the Lucene index and are processed by cleaning the text. The proper nouns that have occurred in these tweets are determined using Stanford POS tagger[3] along with their frequency of occurrence in the tweets. Frequent bi-gram nouns are also extracted and then using the episode detection algorithm, all the episodes that have occurred over the lifetime of the entity are detected.

The following are the conditions to be satisfied to say that an episode has occurred on a short duration of time:

Table 1: Episodes detected of 'Federer'

Rank	Episode	Date/Duration	Maximum Frequent Tweet [[Frequent Nouns]]	Frequency [[Tweet Spike]]	*Related Web URL [1]
3	Entering into Wimbledon '12 finals	07/06/12 to 07/07/12	RT @Wimbledon: Federer will get a crack at his 7th #Wimbledon title beating Djokovic 6-3 3-6 6-4 6-3 to reach Sunday's final. http://t.c ... [[Wimbledon, Federer, crack, Djokovic, title, Sunday]]	3464 [[22094]]	http://www.bbc.co.uk/sport/0/tennis/18740443
1	Winning Wimbledon '12 title	07/08/12 to 07/10/12	RT @AndrewBloch: In 2003 a man predicted Federer would win 7 Wimbledon titles. He died in 2009 and left the bet to charity. Today Oxfam ... [[Federer, Wimbledon, title, man, Murray, today, bet, charity]]	6230 [[48919]]	http://www.atpworldtour.com/News/Tennis/2012/07/27/Wimbledon-Sunday2-Final-Report.aspx
5	Blog on Murray and Federer in Finals	07/21/12	RT @CrowdedSounds: Fan of both Federer and Murray? http://t.co/eOeQjSbu [[Fan, Federer, Murray]]	1636 [[7693]]	http://t.co/eOeQjSbu
2	About Federer	08/03/12 to 08/05/12	RT @Persie_Official: Federer is the boss [[Federer, gold, Andy, Murray, Wimbledon, mens, singles]]	3360 [[39646]]	–
4	Federer's Birthday	08/08/12	RT @ATPWorldTour: Roger #Federer turns 31 today! Retweet to wish him a happy birthday! #atp #tennis [[Federer, Roger, Birthday, Today, retweet]]	2180 [[10832]]	http://www.tennisnow.com/News/Happy-Birthday-Mr–Federer.aspx
6	Winning Cincy Tennis title	08/19/12	RT @ATPWorldTour: #Federer beats @DjokerNole 60 76(7) to win fifth @CincyTennis crown, ties @RafaelNadala's record 21 Masters 1000 titles ... [[Roger, Federer, Cincinnati, Masters, title, congrats, today, Djokovic]]	722 [[6022]]	http://www.espn.co.uk/tennis/sport/story/165924.html

1) The total number of tweets that are related to the event considering retweet count should be greater than *minNumTweets* (parameter).

$$T_E >= minNumTweets \tag{1}$$

where T_E is the total number of tweets that are related to event E.

2) For each day, spike extent (*spikeExtent*) is calculated. Let the day be represented by d and D is the number of days in the lifetime of given entity. The number of tweets related to the event E on a day d are *NumTweets(d,E)*

$$spikeExtent(d, E) = NumTweets(d, E) - NumTweets(d-1, E) \tag{2}$$

$$\max_{d=0}^{d=D}(spikeExtent(d, E)) >= spikeLimit \tag{3}$$

whereas

$$spikeLimit = T_E/spikeFactor \tag{4}$$

spikeFactor (0 < spikeFactor <= T_E) is set manually. The maximum *spikeExtent* of all days should be greater than the *spikeLimit* threshold. The number of days the *spikeExtent* is greater than the *spikeLimit* is also counted as *spikeFreq*. The day on which the *spikeExtent* is maximum is the *spikeDay*.

3) The tweets on *spikeDay* are processed and then all the nouns in those tweets are extracted along with their occurrence frequency in the tweets. If the maximum frequent nouns which are most frequent after the query words corresponds to a single or at most two topics then the event is an *Episode*.

The difference between the number of tweets on a particular day and the number of tweets of the previous day is calculated for each day and the days are sorted in decreasing order based on this difference that is computed. The days which also satisfy the above conditions are considered as *spikeDays*.

The following additional information is extracted for each episode:

1) Let $Freq_N$, $Freqrt_N$ are arrays of nouns which are stored in decreasing order of their frequency from the tweets without and with retweet count correspondingly on the *spikeDay*. First 20 elements of $Freq_N$ and $Freqrt_N$ are extracted.

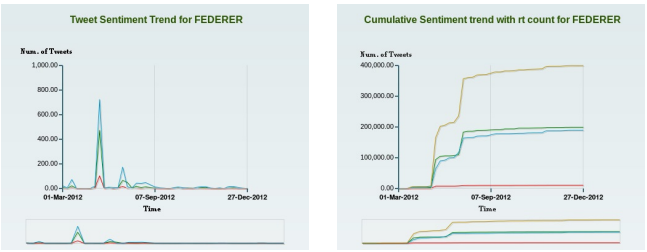
2) Let $Freq_B$, $Freqrt_B$ are arrays of bigram nouns which are stored in the decreasing order of their frequency from

[1]Note: * - not generated from our algorithm, but provided by us as a verification of the episode detected.

the tweets without and with retweet count correspondingly on the *spikeDay*. First 50 elements of $Freq_B$ and $Freqrt_B$ are extracted. Similarly, let us say $FreqPos_B$, $FreqNeg_B$ and $FreqNeu_B$ are arrays with bigrams which are extracted from tweets with positive, negative and neutral sentiments on the *spikeDay* correspondingly. First 50 elements from each of $FreqPos_B$, $FreqNeg_B$, $FreqNeu_B$ are also extracted.

3) Let *Tmax* is the tweet which has maximum retweet count on the *spikeDay* and *Tnoun* is array of nouns present in *Tmax*. *Tmax* is extracted and *Tnoun* is determined from *Tmax*. In addition to the above, the difference between maximum retweet count and minimum retweet count of the tweet on the *spikeDay* (*MaxMindiff*) is also extracted.

Figures 3.(a), 3.(b): Sentiment Trends of 'Federer'



From the tweets, all the above information is extracted and then top k (can be set manually) of the nouns, bigrams and the maximum frequent tweet, nouns in that tweet are all presented in the results as episodes.

4.2 Episode Analytics on Tweets

As a part of episode analytics for twitter, the sentiment trend and cumulative trend of tweets with retweet count are also presented as charts. Number of tweets with different polarities in each 100 tweets are also shown. For all the episodes their strength is calculated and presented in a chart. A chart with all the episodes of entity is generated and presented.

For an entity that has been given as input, until a maximum of 10 episodes are detected based on the threshold and the number of tweets related to the entity. The episodes are ranked based on their strengths. The strength of an episode is calculated as the ratio of the number of tweets that are tweeted about it and the time period over which the episode has occurred. The strength is the average number of tweets that are tweeted per day in the duration of the episode. The formula of the strength is given below:

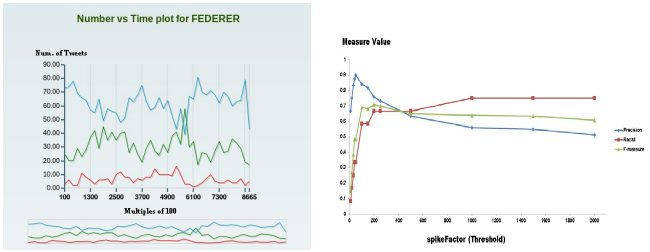
$$S_E = (\sum_{i=1}^{n} N_i)/n \tag{5}$$

where S_E is the Strength of an Episode (E) and N_i is the number of tweets on i^{th} day where as n is the number of days the episode has occurred.

The episodes are further sorted based on the time of their occurrence and all the episodes are presented from the start to the end of the lifetime of the entity. For us, the start and end times are the start and end points of the tweet collection.

Apart from the episode detection, the trends or patterns in the number of tweets and their sentiments are visualized. Basic polarity scoring algorithm is implemented by using

Figure 4.(a): Sentiment Trends of 'Federer' and Figure 4.(b): Thresholds measures chart



cumulative polarity of adjectives. It is explained below in brief.

4.3 Sentiment Analysis

Given a piece of text, sentiment analysis algorithm will give the sentiment score of the text. The text is split by sentence and then all the words like stop words and others that has no sentiment or opinion in it are removed. The list of stop words used is taken from the Stanford stop word list[4] Sentiment lexicon has a list of words with their polarity score. It is taken from MPQA Subjectivity Lexicon[2]. The polarity score of the remaining words from the sentence which are present in the sentiment lexicon are added, which adds upto polarity score of a sentence. The polarity scores of all the sentences in the text are added to get the sentiment score of the total text. The sentiment score can be either positive, zero or negative, depending upon whether the text has positive opinion, neutral opinion or negative opinion.

5. RESULTS AND EVALUATION

In this section, we evaluate the proposed episode detection method by analysing the episodes strength for some famous personalities(entities). We have considered the twitter data from March 2012 to December 2012 for our experiments, so the episodes detected will fall into this timeline.

We have experimented with some queries like "Federer", "Serena Williams", "Lumia 920". We will be analysing the results on the entity query "Federer" in this section. Our Episode Detection algorithm has found 6 episodes related to "Federer" over the period of consideration(March '12 to December '12) and they are presented in Table 1 in sorted order of time.

Each Episode in the table has the following fields: Rank of the episode, episode description, date/duration of the episode, Maximum Frequent Tweet during the episode and Frequent Nouns, Frequency of the maximum frequent tweet and tweet spike, finally the web URL which shows details of the episode on the internet.

The rank of the episode is decided based on the strength of the episode that is being calculated. Episode description is the description in short for the episode that is detected. Date/duration of an episode is the period in which the episode has occurred. Maximum Frequent Tweet is the tweet which have occurred maximum number of times in the episode time period and Frequent Nouns are the nouns that are related to the episode which are sorted based on their frequency of occurrence. Frequency is the number of times the tweet has occurred where as tweet spike is the total number of tweets that are tweeted in the duration of the

episode. For evaluating the episode that is detected, we have searched on the internet and then included the web URL of the page which shows the details of an episode and so proving the occurrence of that corresponding episode. Observe that the dates of the articles in the web URLs are same as the dates of occurrence of its corresponding episode. Each of the episode detected related to "Federer" is analysed further based on their date of occurrence below:

1) **The first episode** has occurred on 6th and 7th of July 2012 when Federer won the semi finals against Djokovic and entered into Wimbledon '12 Finals just before the day of the finals. The rank of this episode is 3 and the maximum frequent tweet has tweeted 3464 times. The frequent nouns are wimbledon, federer, crack, Djokovic, title, sunday. The web URL shows that Federer has entered into finals by winning over Djokovic dated 6th of July 2012.

2) **The second episode** is after Federer winning the Wimbledon '12 Finals over Murray. This episode is ranked number 1 and has occurred between 8th and 10th July 2012. Maximum frequent tweet has been tweeted 6230 times. Federer, wimbledon, title, man, murray, today are frequent nouns. The web page talks about Federer winning Wimbledon for the 7th time.

3) **The third episode** is the blog that is written about the final match between Federer and Murray and how people want both to win the match. This episode has occurred on 21st July 2012, 9days after the blog has been posted. Frequent nouns are fan, federer, murray. This might be because this is not an event, but the opinion of a person written in the form of a blog and so it took time to tweak. It is number 5 episode and the tweet itself has the URL to the blog.

4) **Robin Van Persie** tweets about Federer. Many people have retweeted it as they share the same opinion and so this has become an episode. The rank is 2 and this tweet has retweeted 3360 times. Federer, gold, Andy, Murray, wimbledon, mens, singles are frequent nouns.

5) **Federer's 31st Birthday** is the fifth episode that has occurred on his birthday 8th August 2012. It is rank 4 and 2180 people has tweeted the same birthday wishes tweet to "Federer". Frequent nouns are Roger, Federer, birthday, today.

6) **The last episode** is about Federer winning the Cincy Tennis Crown on 19th August 2012. Frequent nouns are Roger, Federer, Cinnicati, masters, title, congrats, today. The episode is ranked 6 and the url shows details about the episode.

All these episodes are sorted and their strengths are calculated and then the episodes strength of the entity is generated. The chart in figure 2 shows the strength of detected episodes of "Federer" with Time on X-axis and Number of days an episode has occurred on Y-axis. The radius of the bubble is taken as the strength of an episode. The strength is divided by 50000 to mark it as radius just to scale the value to fit into the chart.

Figures 3.(a), 3.(b) and 4.(a) shows the sentiment trends of tweets related to "Federer" over the time line. The sentiment trends charts are generated using Zingchart javascript library[5](free branded version). Figure 3.(a) shows the number of tweets that are tweeted positive (green line), negative (red line) or neutral (blue line) with sentiment on each day. Figure 3.(b) shows the number of tweets that are tweeted positive (green line), negative (red line), neutral (blue line) with sentiment or all in total (yellow line) until that day from the start day with retweet count. We can see there is a sudden spike in the number of tweets at several places. Figure 4.(a) shows the number of positive (green line), negative (red line) and neutral (blue line) tweets with sentiment that are present in every 100 tweets.

The episodes of "Narendra Modi" were also detected. "Narendra Modi" is an Indian Politician, Chief Minister of the state Gujarat in India. Table 2 shows episodes detected for the entity "Narendra Modi" with 6 episodes presented based on their occurrence date.

A brief analyis of the episodes detected is done below based on their date of occurrence: 1) The rank of the first episode is 1 and it occurred on 03/17/12. The episode is Modi on cover page of Time Magazine. 2) This episode occurred on 07/24/12 about Modi going to Japan. The rank of the episode is 3. 3) This episode is Modi wishing everyone on Janmastami. The rank of this episode is 4 and occurred on 08/10/12. 4) The episode with rank 6 has occurred on Modi's Birthday on 09/17/12. 5) The episode occurred after Modi completed 4000 days as Gujarat's CM and the rank of the episode is 5. It has occurred on 09/18/12. 6) Message from Modi is the next episode whose rank is 2. It has occurred on 10/13/12.

As a part of TEA system evaluation, we have calculated precision, recall and F-measure of our TEA approach. For an entity, the detected episodes are classified manually to be either valid or invalid episodes. An episode is valid if it is a significant event that has occurred in the lifespan of that particular entity. The ratio of number of episodes that are valid to the total number of episodes detected will be the precision of our TEA algorithm for that particular entity. The precision of TEA system is calculated by taking the average precision of all the entities.

The recall of TEA system for a particular entity is the ratio of number of valid episodes to the actual number of episodes that have occurred over that entity's lifespan in twitter. The recall of our TEA algorithm is the average recall of all the entities. However, it is difficult to determine how many episodes have actually occurred for an entity over its twitter lifespan. So, for each entity we have manually searched over the internet (mostly their Wikipedia pages) and listed down the significant events that have occurred over a period from March 2012 to December 2012.

Table 3 shows the precision and recall for each entity that is given as input to the TEA system. The overall precision of the system that is calculated over these 11 entities is 0.864 whereas the overall recall of the system is 0.503.

F1-score (F-measure) is a measure of a test's accuracy. The F1-score can be interpreted as a weighted average of the precision and recall and it's formula is given by:

$$F1\text{-}score = 2 * (Precision * Recall)/(Precision + recall) \quad (6)$$

Table 4 shows the F1-score (f-measure) that are computed using precision and recall values from table 3 for each of the entities that are considered. The overall F1 score of TEA system is 0.62.

The precision, recall and f-measure values that are presented for different entities are calculated by setting different thresholds (*spikeFactor*) for different entities. These validation measure values change based on the threshold value that is set. For entity 'Narendra Modi', we have presented values of validation measures for different thresholds. Fig-

Table 2: Episodes of 'Narendra Modi' over its lifespan

Rank	Episode	Date/ Duration	Maximum Frequent Tweet [[Frequent Nouns]] [[Polarity Score]]	Frequency	*Related Web URL [2]
1	Modi on cover page of Times Magazine	03/17/12	RT @vijsimha: Here's news more interesting than #Budget2012. Time magazine puts Narendra Modi on cover as the man who could change Indi ... [[news, time, magazine, narendra, modi, cover, man]] [[1]]	314	http://timesofindia.indiatimes.com/india/Narendra-Modi-on-Time-magazine-cover/articleshow/12296366.cms
3	Modi going to Japan	07/24/12	RT @sardesairajdeep: Appreciate Narendra Modi for going to Japan and standing by Haryana govt. Nation above politics. (there you go folk ... [[narendra, modi, japan, standing, haryana, govt]] [[1]]	280	http://articles.economictimes.indiatimes.com/2012-07-23/news/32804624_1_maruti-suzuki-s-manesar-manesar-plant-maruti-s-manesar
4	Modi wishes on Janmashtami	08/10/12	RT @TOIBlogs: Janmashtami the protector of cows, Lord Krishna's birthday : Narendra Modi http://t.co/foHZ8Qwb [[protector, cow, lord, krishna, birthday, narendra]] [[1]]	86	http://t.co/foHZ8Qwb
6	Modi's Birthday	09/17/12	RT @Ohfakenews: Narendra Modi turns 62 today. You may remember him from his biggest hit: Naroda Patiya riots. #HappyBdayNamo #NaMo [[narendra, modi, today, hit, #happybdaynamo, #namo]] [[0]]	27	http://en.wikipedia.org/wiki/Narendra_Modi
5	4000 days as Gujarat's CM	09/18/12	RT @sardesairajdeep: Narendra Modi completes 4000 days as Gujarat chief minister today. Quite an achievement Shouldn't that be trending? [[narendra, modi, days, gujarat, chief, minister, today]] [[0]]	93	http://samvada.org/2012/news/4000-days-as-cm-narendra-modi-takes-gujarat-as-model-state-of-india-in-development/
2	Message from Modi	10/13/12	RT @Swamy39: Narendra Modi: UK has melted. US is not far behind. The hidden message is that if we are strong then they will come looking ... [[narendra, modi, message]] [[1]]	437	-

Table 3: Precision and Recall of Entities

Entity (query)	Precision	Recall	Entity (query)	Precision	Recall
Narendra Modi	0.9	0.333	Federer	1	0.588
Barack Obama	0.9	0.642	Britney Spears	0.8	0.4
Sachin	1	0.5	Serena Williams	1	0.83
Adele	0.5	0.5	Andy Murray	0.7	0.571
Life of Pi	0.9	0.33	Lumia 920	1	0.33
Taylor Swift	0.8	0.5			

Table 4: F-measure values of Entities

Entity (query)	F1 score	Entity (query)	F1 score
Narendra Modi	0.486	Federer	0.740
Barack Obama	0.749	Britney Spears	0.533
Sachin	0.667	Serena Williams	0.907
Adele	0.5	Andy Murray	0.629
Life of Pi	0.486	Lumia 920	0.499
Taylor Swift	0.615		

ure 4.(b) shows how precision, recall and f-measure values change with *spikeFactor* (threshold). The plot shows precision, recall and f-measure values on Y-axis for different thresholds on X-axis. The blue line in the plot corresponds to precision, maroon line corresponds to recall and green line to F-measure. The precision started low, increased to a maximum value and then decreased with increase in *spikeFactor*. Whereas, the recall started even low and increased

[2]Note: * - not generated from our algorithm, but provided by us as a verification of the episode detected.

Table 5: Validation measures for different thresholds of 'Narendra Modi'

spikeFactor (Threshold)	Number of Episodes Detected	Precision	Recall	F1 score
10	3	0.67	0.08	0.15
20	4	0.75	0.17	0.27
30	6	0.83	0.25	0.38
40	9	0.88	0.33	0.48
50	9	0.9	0.33	0.49
100	20	0.84	0.58	0.69
150	23	0.82	0.58	0.68
200	30	0.76	0.67	0.71
250	39	0.73	0.67	0.7
500	61	0.63	0.67	0.65
1000	85	0.56	0.75	0.64
1500	105	0.55	0.75	0.63
2000	120	0.51	0.75	0.61

with *spikeFactor* until it reached a maximum value and then it became constant from there. F-measure followed a similar pattern as that of precision curve. Table 5 shows the precision, recall and f-measure values for different *spikeFactor*.

The top 6 episodes that are detected for entity 'Narendra Modi' when threshold (*spikeFactor*) is set to be 50 are presented in Table 2 and validation measures for different thresholds for 'Narendra Modi' are presented in Table 5.

6. CONCLUSIONS

Our intention to infer significant knowledge/insight from huge number of tweets raises problems. The key issue is to comprehend what a set of tweets convey about an entity. Our approach has been to consider lifetime of an entity and determine what all events can occur in it. From the events one can get episodes that convey larger description of the set of tweets are conveying, and then episodes strength of an entity are shown. We built a system for taking any entity as a keyword and process relevant tweets to detect the episodes. Our results validate our approach by providing episodes that provide the essence of information that can be gleaned from tweets. In particular, we are able to convey sentiments about tweets and phrases that describe tweets over different periods of time. Therefore, our system can be used to determine short term understanding from tweets about a given entity and use it to promote or rectify certain actions. For example, sell more mobile phones at discount or quickly send out a patch for a malfunctioning applet. As part of future work we will continue to improve core algorithms applied in this paper, and delve into what can be learned from detected episodes.

References

[1] *Apache Lucene.* https://lucene.apache.org/.

[2] *MPQA Subjectivity Lexicon.* http://mpqa.cs.pitt.edu/.

[3] *Stanford Part-Of-Speech Tagger.* http://nlp.stanford.edu/software/tagger.shtml.

[4] *Stanford Stop-Word List.* http://www.wordsift.com/wordlists.

[5] *ZingChart Javascript Charting Library.* http://www.zingchart.com.

[6] S. Asur and B. A. Huberman. Trends in social media: Persistence and decay. *AAAI*, 2011.

[7] H. Becker and M. Naaman. Beyond trending topics: Real-world event identification on twitter. *AAAI*, 2011.

[8] M. S. Bernstein and B. S. Eddi. Interactive topic-based browsing of social status streams. *UIST*, 2010.

[9] D. Gruhl and R. Guha. Information diffusion through blogspace. *WWW*, 2004.

[10] M. Iwata and T. Saka. Aspectiles: Tile-based visualization of diversified web search results. *SIGIR*, 2012.

[11] H. Kwak and C. Lee. What is twitter, a social network or a news media? *WWW*, 2010.

[12] M. Mathioudakis and N. Koudas. Twittermonitor: Trend detection over the twitter stream. *SIGMOD*, 2010.

[13] M. R. Morris and S. Counts. Tweeting is believing? understanding microblog credibility perceptions. *CSCW*, 2012.

[14] J. Nichols and J. Mahmud. Summarizing sporting events using twitter. *ACM IUI*, 2012.

[15] T. Sakaki and M. Okazaki. Earthquake shakes twitter users: real-time event detection by social sensors. *WWW*, 2010.

[16] Teevan and Ramage. Twittersearch: A comparison of microblog search and web search. *WSDM*, 2011.

Sentic API

A Common-Sense Based API for Concept-Level Sentiment Analysis

http://sentic.net/api

Erik Cambria
Temasek Laboratories
National University of Singapore
cambria@nus.edu.sg

Soujanya Poria
School of Electrical & Electronic Engineering
Nanyang Technological University
sporia@ntu.edu.sg

Alexander Gelbukh
Center for Computing Research
National Polytechnic Institute of Mexico
gelbukh@cic.ipn.mx

Kenneth Kwok
Temasek Laboratories
National University of Singapore
kenkwok@nus.edu.sg

ABSTRACT

The bag-of-concepts model can represent semantics associated with natural language text much better than bags-of-words. In the bag-of-words model, in fact, a concept such as `cloud_computing` would be split into two separate words, disrupting the semantics of the input sentence. Working at concept-level is important for tasks such as opinion mining, especially in the case of microblogging analysis. In this work, we present Sentic API, a common-sense based application programming interface for concept-level sentiment analysis, which provides semantics and sentics (that is, denotative and connotative information) associated with 15,000 natural language concepts.

Categories and Subject Descriptors

H.3.1 [**Information Systems Applications**]: Linguistic Processing; I.2.7 [**Natural Language Processing**]: Language parsing and understanding

General Terms

Algorithms

Keywords

Natural language processing; Sentiment analysis

1. INTRODUCTION

Hitherto, online information retrieval, aggregation, and processing have mainly been based on algorithms relying on the textual representation of webpages. Such algorithms are very good at retrieving texts, splitting them into parts, checking the spelling and counting the number of words.

Published as part of the #Microposts2014 Workshop proceedings, available online as CEUR Vol-1141 (`http://ceur-ws.org/Vol-1141`)

#Microposts2014, April 7th, 2014, Seoul, Korea.

But when it comes to interpreting sentences and extracting meaningful information, their capabilities are known to be very limited. Machine-learning algorithms, in fact, are limited by the fact that they can process only the information that they can 'see'. As human text processors, we do not have such limitations as every word we see activates a cascade of semantically related concepts, relevant episodes, and sensory experiences, all of which enable the completion of complex tasks – such as word-sense disambiguation, textual entailment, and semantic role labeling – in a quick and effortless way. Machine learning techniques, moreover, are intrinsically meant for chunking numerical data. Through escamotages such as word frequency counting, it is indeed possible to apply such techniques also in the context of natural language processing (NLP), but it would be no different from trying to understand an image by solely looking at bits per pixel information.

Concept-level sentiment analysis, instead, focuses on a semantic analysis of text through the use of web ontologies or semantic networks, which allow the aggregation of conceptual and affective information associated with natural language opinions. By relying on large semantic knowledge bases, such approaches step away from blind use of keywords and word co-occurrence count, but rather rely on the implicit features associated with natural language concepts. Unlike purely syntactical techniques, concept-based approaches are able to detect also sentiments that are expressed in a subtle manner, e.g., through the analysis of concepts that do not explicitly convey any emotion, but which are implicitly linked to other concepts that do so. The bag-of-concepts model can represent semantics associated with natural language much better than bags-of-words. In the bag-of-words model, in fact, a concept such as `cloud_computing` would be split into two separate words, disrupting the semantics of the input sentence (in which, for example, the word `cloud` could wrongly activate concepts related to `weather`).

By allowing for the inference of semantics and sentics, the analysis at concept-level enables a comparative fine-grained feature-based sentiment analysis. Rather than gathering isolated opinions about a whole item (e.g., iPhone 5S or Galaxy S5), users are generally more interested in comparing different products according to their specific features (e.g., iPhone 5S's vs Galaxy S5's touchscreen), or sub-features (e.g., fragility of iPhone 5S's vs Galaxy S5's touchscreen). In this context, the construction of comprehensive common and common-sense knowledge bases is key for feature-spotting and polarity detection, respectively.

Common-sense, in particular, is necessary to properly deconstruct natural language text into sentiments– for example, to appraise the concept `small_room` as negative for a hotel review and `small_queue` as positive for a post office, or the concept `go_read_the_book` as positive for a book review but negative for a movie review.

The rest of the paper is organized as follows: Section 2 presents available resources for concept-level sentiment analysis; Section 3 illustrates the techniques exploited to build the Sentic API; Section 4 describes in detail how the API is developed and how it can be used; Section 5 proposes an evaluation of the API; finally, Section 6 concludes the paper and suggests further research directions.

2. RELATED WORK

Commonly used resources for concept-level sentiment analysis include ANEW [3], WordNet-Affect (WNA) [21], ISEAR [1], SentiWordNet [9], and SenticNet [7]. In [22], for example, a concept-level sentiment dictionary is built through a two-step method combining iterative regression and random walk with in-link normalization. ANEW and SenticNet are exploited for propagating sentiment values based on the assumption that semantically related concepts share common sentiment. Moreover, polarity accuracy, Kendall distance, and average-maximum ratio are used, in stead of mean error, to better evaluate sentiment dictionaries.

A similar approach is adopted in [19], which presents a methodology for enriching SenticNet concepts with affective information by assigning an emotion label to them. Authors use various features extracted from ISEAR, as well as similarity measures that rely on the polarity data provided in SenticNet (those based on WNA) and ISEAR distance-based measures, including point-wise mutual information, and emotional affinity. Another recent work that builds upon an existing affective knowledge base is [14], which proposes the re-evaluation of objective words in SentiWordNet by assessing the sentimental relevance of such words and their associated sentiment sentences. Two sampling strategies are proposed and integrated with support vector machines for sentiment classification. According to the experiments, the proposed approach significantly outperforms the traditional sentiment mining approach, which ignores the importance of objective words in SentiWordNet. In [2], the main issues related to the development of a corpus for opinion mining and sentiment analysis are discussed both by surveying the existing work in this area and presenting, as a case study, an ongoing project for Italian, called Senti-TUT, where a corpus for the investigation of irony about politics in social media is developed.

Other work explores the ensemble application of knowledge bases and statistical methods. In [24], for example, a hybrid approach to combine lexical analysis and machine learning is proposed in order to cope with ambiguity and integrate the context of sentiment terms. The context-aware method identifies ambiguous terms that vary in polarity depending on the context and stores them in contextualized sentiment lexicons. In conjunction with semantic knowledge bases, these lexicons help ground ambiguous sentiment terms to concepts that correspond to their polarity.

More machine-learning based works include [10], which introduces a new methodology for the retrieval of product features and opinions from a collection of free-text customer reviews about a product or service. Such a methodology relies on a language modeling framework that can be applied to reviews in any domain and language provided with a seed set of opinion words. The methodology combines both a kernel-based model of opinion words (learned from the seed set of opinion words) and a statistical mapping between words to approximate a model of product features from which the retrieval is carried out.

3. TECHNIQUES ADOPTED

In this work, we exploit the ensemble application of spectral association [12], an approximation of many steps of spreading activation, and CF-IOF (concept frequency - inverse opinion frequency), an approach similar to TF-IDF weighting, to extract semantics from ConceptNet [20], a semantic network of common-sense knowledge. The extraction of sentics, in turn, is performed through the combined use of AffectiveSpace [4], a multi-dimensional vector space representation of affective common-sense knowledge, and the Hourglass of Emotions [6], a brain-inspired emotion categorization model.

3.1 Spectral Association

Spectral association is a technique that involves assigning activations to 'seed concepts' and applying an operation that spreads their values across the graph structure of ConceptNet. This operation transfers the most activation to concepts that are connected to the key concepts by short paths or many different paths in common-sense knowledge.

In particular, we build a matrix C that relates concepts to other concepts, instead of their features, and add up the scores over all relations that relate one concept to another, disregarding direction. Applying C to a vector containing a single concept spreads that concept's value to its connected concepts. Applying C^2 spreads that value to concepts connected by two links (including back to the concept itself). As we aim to spread the activation through any number of links, with diminishing returns, the operator we want is:

$$1 + C + \frac{C^2}{2!} + \frac{C^3}{3!} + ... = e^C$$

We can calculate this odd operator, e^C, because we can factor C. C is already symmetric, so instead of applying Lanczos' method [15] to CC^T and getting the singular value decomposition (SVD), we can apply it directly to C and get the spectral decomposition $C = V\Lambda V^T$. As before, we can raise this expression to any power and cancel everything but the power of Λ. Therefore, $e^C = Ve^{\Lambda}V^T$. This simple twist on the SVD lets us calculate spreading activation over the whole matrix instantly. We can truncate this matrix to k axes and therefore save space while generalizing from similar concepts. We can also rescale the matrix, so that activation values have a maximum of 1 and do not tend to collect in highly-connected concepts, by normalizing the truncated rows of $Ve^{\Lambda/2}$ to unit vectors, and multiplying that matrix by its transpose to get a rescaled version of $Ve^{\Lambda}V^T$.

3.2 CF-IOF Weighting

CF-IOF is a technique that identifies common topic-dependent semantics in order to evaluate how important a concept is to a set of opinions concerning the same topic. It is hereby used to feed spectral association with 'seed concepts'. Firstly, the frequency of a concept c for a given domain d is calculated by counting the occurrences of the concept c in the set of available d-tagged opinions and dividing the result by the sum of number of occurrences of all concepts in the set of opinions concerning d. This frequency is then multiplied by the logarithm of the inverse frequency of the concept in the whole collection of opinions, that is:

$$CF\text{-}IOF_{c,d} = \frac{n_{c,d}}{\sum_k n_{k,d}} \log \sum_k \frac{n_k}{n_c}$$

where $n_{c,d}$ is the number of occurrences of concept c in the set of opinions tagged as d, n_k is the total number of concept occurrences and n_c is the number of occurrences of c in the whole set of opinions.

A high weight in CF-IOF is reached by a high concept frequency (in the given opinions) and a low opinion frequency of the concept in the whole collection of opinions. Therefore, thanks to CF-IOF weights, it is possible to filter out common concepts and detect relevant topic-dependent semantics.

3.3 AffectiveSpace

To extract sentics from natural language text, we use AffectiveSpace, a multi-dimensional vector space built upon ConceptNet and WNA. The alignment operation operated over these two knowledge bases yields a matrix, A, in which common-sense and affective knowledge coexist, i.e., a matrix 15,000 $\times$ 118,000 whose rows are concepts (e.g., `dog` or `bake_cake`), whose columns are either common-sense and affective features (e.g., *isA-pet* or *hasEmotion-joy*), and whose values indicate truth values of assertions.

Therefore, in A, each concept is represented by a vector in the space of possible features whose values are positive for features that produce an assertion of positive valence (e.g., 'a penguin is a bird'), negative for features that produce an assertion of negative valence (e.g., 'a penguin cannot fly') and zero when nothing is known about the assertion. The degree of similarity between two concepts, then, is the dot product between their rows in A. The value of such a dot product increases whenever two concepts are described with the same feature and decreases when they are described by features that are negations of each other. In particular, we use truncated SVD [23] in order to obtain a new matrix containing both hierarchical affective knowledge and common-sense.

The resulting matrix has the form $\tilde{A} = U_k \, \Sigma_k \, V_k^T$ and is a low-rank approximation of A, the original data. This approximation is based on minimizing the Frobenius norm [13] of the difference between A and $\tilde{A}$ under the constraint $rank(\tilde{A}) = k$. For the Eckart–Young theorem [8] it represents the best approximation of A in the least-square sense, in fact:

$$\begin{aligned} \min_{\tilde{A}|rank(\tilde{A})=k} |A - \tilde{A}| &= \min_{\tilde{A}|rank(\tilde{A})=k} |\Sigma - U^* \tilde{A} V| \\ &= \min_{\tilde{A}|rank(\tilde{A})=k} |\Sigma - S| \end{aligned}$$

assuming that $\tilde{A}$ has the form $\tilde{A} = USV^*$, where S is diagonal. From the rank constraint, i.e., S has k non-zero diagonal entries, the minimum of the above statement is obtained as follows:

$$\min_{\tilde{A}|rank(\tilde{A})=k} |\Sigma - S| = \min_{s_i} \sqrt{\sum_{i=1}^{n} (\sigma_i - s_i)^2} =$$

$$= \min_{s_i} \sqrt{\sum_{i=1}^{k} (\sigma_i - s_i)^2 + \sum_{i=k+1}^{n} \sigma_i^2} = \sqrt{\sum_{i=k+1}^{n} \sigma_i^2}$$

Therefore, $\tilde{A}$ of rank k is the best approximation of A in the Frobenius norm sense when $\sigma_i = s_i$ $(i = 1, ..., k)$ and the corresponding singular vectors are the same as those of A. If we choose to discard all but the first k principal components, common-sense concepts and emotions are represented by vectors of k coordinates: these coordinates can be seen as describing concepts in terms of 'eigenmoods' that form the axes of AffectiveSpace, i.e., the basis e_0,...,e_{k-1} of the vector space. For example, the most significant eigenmood, e_0, represents concepts with positive affective valence. That is, the larger a concept's component in the e_0 direction is, the more affectively positive it is likely to be. Thus, by exploiting the information sharing property of truncated SVD, concepts with the same affective valence are likely to have similar features – that is, concepts conveying the same emotion tend to fall near each other in AffectiveSpace.

Concept similarity does not depend on their absolute positions in the vector space, but rather on the angle they make with the origin. For example we can find concepts such as `beautiful_day`, `birthday_party`, `laugh` and `make_person_happy` very close in direction in the vector space, while concepts like `sick`, `feel_guilty`, `be_laid_off` and `shed_tear` are found in a completely different direction (nearly opposite with respect to the centre of the space).

3.4 The Hourglass of Emotions

To reason on the disposition of concepts in AffectiveSpace, we use the Hourglass of Emotions (Figure 1), an affective categorization model developed starting from Plutchik's studies on human emotions [18]. In the model, sentiments are reorganized around four independent dimensions whose different levels of activation make up the total emotional state of the mind. The Hourglass of Emotions, in fact, is based on the idea that the mind is made of different independent resources and that emotional states result from turning some set of these resources on and turning another set of them off [16].

The primary quantity we can measure about an emotion we feel is its strength. But when we feel a strong emotion it is because we feel a very specific emotion. And, conversely, we cannot feel a specific emotion like 'fear' or 'amazement' without that emotion being reasonably strong. Mapping this space of possible emotions leads to an hourglass shape.

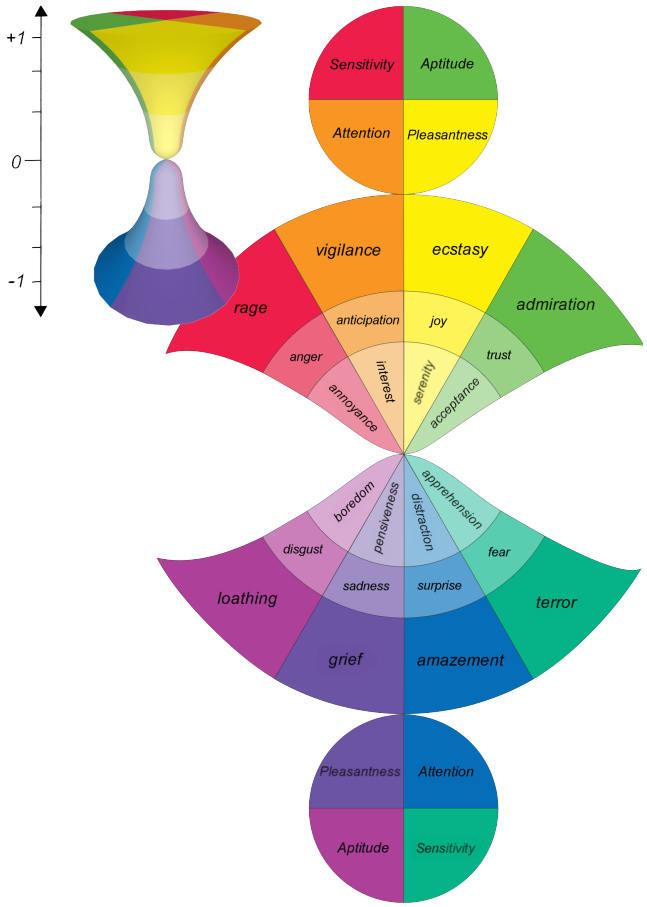


Figure 1: The Hourglass model

In the model, affective states are not classified, as often happens in the field of emotion analysis, into basic emotional categories, but rather into four concomitant but independent dimensions, characterized by six levels of activation, which determine the intensity of the expressed/perceived emotion as a $float \in$ [-1,+1]. Such levels are also labeled as a set of 24 basic emotions (six for each of the affective dimensions) in a way that allows the model to specify the affective information associated with text both in a dimensional and in a discrete form.

4. BUILDING AND USING THE API

Currently available lexical resources for opinion polarity and affect recognition such as SentiWordNet or WNA are known to be pretty noisy and limited. These resources, in fact, mainly provide opinion polarity and affective information at syntactical level, leaving out polarity and affective information for common-sense knowledge concepts like `celebrate_special_occasion`, `accomplish_goal`, `bad_feeling`, `be_on_cloud_nine`, or `lose_temper`, which are usually found in natural language text to express viewpoints and affect.

In order to build a comprehensive resource for opinion mining and sentiment analysis, we use the techniques described in Section 3 to extract both cognitive and affective information from natural language text in a way that it is possible to map it into a fixed structure. In particular, we propose to bridge the cognitive and affective gap between word-level natural language data and their relative concept-level opinions and sentiments, by building semantics and sentics on top of them (Figure 2). To this end, the Sentic API provides polarity (a float number between -1 and +1 that indicates whether a concept is positive or negative), semantics (a set of five semantically-related concepts) and sentics (affective information in terms of the Hourglass affective dimensions) associated with 15,000 natural language concepts. This information is encoded in RDF/XML using the descriptors defined by Human Emotion Ontology (HEO) [11].

4.1 Extracting Semantics

The extraction of semantics associated with common-sense knowledge concepts is performed through the ensemble application of spectral association and CF-IOF on the graph structure of ConceptNet. In particular, we apply CF-IOF on a set of 10,000 topic-tagged posts extracted from LiveJournal[1], a virtual community of more than 23 million who are allowed to label their posts not only with a topic tag but also with a mood label, by choosing from more than 130 predefined moods or by creating custom mood themes.

[1]`http://livejournal.com`

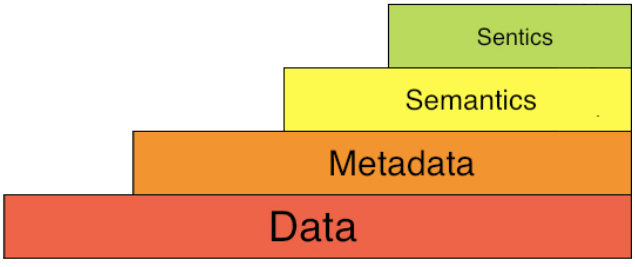


Figure 2: The semantics and sentics stack

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <rdf:Description rdf:about="http://sentic.net/api/concept/celebrate_special_occasion">
        <rdf:type rdf:resource="http://sentic.net/api/concept/semantics"/>
        <semantics rdf:resource="http://sentic.net/api/en/concept/celebrate_holiday"/>
        <semantics rdf:resource="http://sentic.net/api/en/concept/celebrate_occasion"/>
        <semantics rdf:resource="http://sentic.net/api/en/concept/celebrate_birthday"/>
        <semantics rdf:resource="http://sentic.net/api/en/concept/celebrate_wed"/>
        <semantics rdf:resource="http://sentic.net/api/en/concept/express_appreciation"/>
    </rdf:Description>
</rdf:RDF>
```

Figure 3: XML file resulting from querying about the semantics of `celebrate_special_occasion`

Thanks to CF-IOF weights, it is possible to filter out common concepts and detect domain-dependent concepts that individualize topics typically found in online opinions such as art, food, music, politics, family, entertainment, photography, travel, and technology. These concepts represent seed concepts for spectral association, which spreads their values across the ConceptNet graph. In particular, in order to accordingly limit the spreading activation of ConceptNet nodes, the rest of the concepts detected via CF-IOF are given as negative inputs to spectral association so that just domain-specific concepts are selected.

4.2 Extracting Sentics

The extraction of sentics associated with common-sense knowledge concepts is performed through the combined use of AffectiveSpace and the Hourglass model. In particular, we discard all but the first 100 singular values of the SVD and organize the resulting vector space using a k-medoids clustering approach [17], with respect to the Hourglass of Emotions (i.e., by using the model's labels as 'centroid concepts').

By calculating the relative distances (dot product) of each concept from the different centroids, it is possible to calculate its affective valence in terms of Pleasantness, Attention, Sensitivity and Aptitude, which is stored in the form of a four-dimensional vector, called sentic vector.

4.3 Encoding Semantics and Sentics

In order to represent the Sentic API in a machine-accessible and machine-processable way, results are encoded in RDF triples using a XML syntax (Figure 3). In particular, concepts are identified using the ConceptNet Web API and statements are encoded in RDF/XML format on the base of HEO. Statements have forms such as *concept – hasPlesantness – pleasantnessValue, concept – hasPolarity – polarityValue, and concept – isSemanticallyRelatedTo – concept*.

Given the concept `celebrate_special_occasion`, for example, the Sentic API provides a set of semantically related concepts, e.g., `celebrate_birthday`, and a sentic vector specifying Pleasantness, Attention, Sensitivity and Aptitude associated with the concept (which can be decoded into the emotions of *ecstasy* and *anticipation* and from which a positive polarity value can be inferred).

Encoding semantics and sentics in RDF/XML using the descriptors defined by HEO allows cognitive and affective information to be stored in a Sesame triple-store, a purpose-built database for the storage and retrieval of RDF metadata. Sesame can be embedded in applications and used to conduct a wide range of inferences on the information stored, based on RDFS and OWL type relations between data. In addition, it can also be used in a standalone server mode, much like a traditional database with multiple applications connecting to it.

4.4 Exploiting Semantics and Sentics

Thanks to its Semantic Web aware format, the Sentic API is very easy to interface with any real-world application that needs to extract semantics and sentics from natural language. This cognitive and affective information is supplied both at category-level (through domain and sentic labels) and dimensional-level (through polarity values and sentic vectors).

Sentic labels, in particular, are useful in case we deal with real-time adaptive applications (in which, for example, the style of an interface or the expression of an avatar has to quickly change according to labels such as 'excitement' or 'frustration' detected from user input). Polarity values and sentic vectors, in turn, are useful for tasks such as information retrieval and polarity detection (in which it is needed to process batches of documents and, hence, perform calculations, such as addition, subtraction, and average, on both conceptual and affective information).

Averaging results obtained at category-level is also possible by using a continuous 2D space whose dimensions are evaluation and activation, but the best strategy is usually to consider the opinionated document as composed of small bags of concepts (SBoCs) and feed these into the Sentic API to perform statistical analysis of the resulting sentic vectors.

To this end, we use a pre-processing module that interprets all the affective valence indicators usually contained in text such as special punctuation, complete upper-case words, onomatopoeic repetitions, exclamation words, negations, degree adverbs and emoticons, and eventually lemmatizes text.

A semantic parser then deconstructs text into concepts using a lexicon based on 'sentic n-grams', i.e., sequences of lexemes which represent multiple-word common-sense and affective concepts extracted from ConceptNet, WNA and other linguistic resources. We then use the resulting SBoC as input for the Sentic API and look up into it in order to obtain the relative sentic vectors, which we average in order to detect primary and secondary moods conveyed by the analyzed text and/or its polarity, given by the formula [6]:

$$p = \sum_{i=1}^{N} \frac{Plsnt(c_i) + |Attnt(c_i)| - |Snst(c_i)| + Aptit(c_i)}{3N}$$

where N is the size of the SBoC. As an example of how the Sentic API can be exploited for microblogging analysis, intermediate and final outputs obtained when a natural language opinion is given as input to the system can be examined. The tweet "I think iPhone4 is the top of the heap! OK, the speaker is not the best i hv ever seen bt touchscreen really puts me on cloud 9... camera looks pretty good too!" is selected. After the pre-processing and semantic parsing operations, the following SBoCs are obtained:

SBoC#1:
<*Concept: 'think'*>
<*Concept: 'iphone4'*>
<*Concept: 'top heap'*>
SBoC#2:
<*Concept: 'ok'*>
<*Concept: 'speaker'*>
<*Concept:* !*'good'*++>
<*Concept: 'see'*>
SBoC#3:
<*Concept: 'touchscreen'*>
<*Concept: 'put cloud nine'*++>
SBoC#4:
<*Concept: 'camera'*>
<*Concept: 'look good'*−−>

Table 1: Structured output example

Opinion Target	Category	Moods	Polarity
'iphone4'	'phones', 'electronics'	'ecstasy', 'interest'	+0.71
'speaker'	'electronics', 'music'	'annoyance'	-0.34
'touchscreen'	'electronics'	'ecstasy', 'anticipation'	+0.82
'camera'	'photography', 'electronics'	'acceptance'	+0.56

After feeding the extracted concepts to the Sentic API, we can exploit semantics and sentics to detect opinion targets and obtain, for each of these, the relative affective information both in a discrete way (with one or more emotional labels) and in a dimensional way (with a polarity value $\in$ [-1,+1]) as shown in Table 1.

5. EVALUATION

As a use case evaluation of the proposed API, we select the problem of crowd validation of the UK national health service (NHS) [5], that is, the exploitation of the wisdom of patients to adequately validate the official hospital ratings made available by UK health-care providers and NHS Choices[2]. To validate such data, we exploit patient stories extracted from PatientOpinion[3], a social enterprise providing an online feedback service for users of the UK NHS. The problem is that this social information is often stored in natural language text and, hence, intrinsically unstructured, which makes comparison with the structured information supplied by health-care providers very difficult. To bridge the gap between such data (which are different at structure-level yet similar at concept-level), we exploit the Sentic API to marshal PatientOpinion's social information in a machine-accessible and machine-processable format and, hence, compare it with the official hospital ratings provided by NHS Choices and each NHS trust.

In particular, we use Sentic API's inferred ratings to validate the information declared by the relevant health-care providers, crawled separately from each NHS trust website, and the official NHS ranks, extracted using the NHS Choices API[4]. This kind of data usually consists of ratings that associate a polarity value to specific features of health-care providers such as 'communication', 'food', 'parking', 'service', 'staff', and 'timeliness'. The polarity can be either a number in a fixed range or simply a flag (positive/negative).

Since each patient opinion can regard more than one topic and the polarity values associated with each topic are often independent from each other, we need to extract, from each opinion, a set of topics and then, from each topic detected, the polarity associated with it. Thus, after deconstructing each opinion into a set of SBoCs, we analyze these through Sentic API in order to tag each SBoC with one of the relevant topics (if any) and calculate a polarity value. We ran this process on a set of 857 topic- and polarity-tagged short stories extracted from PatientOpinion database and computed recall and precision rates as evaluation metrics.

As for the SBoC categorization, results showed that the Sentic API can detect topics in patient stories with satisfactory accuracy. In particular, the classification of stories about 'food' and 'communication' was performed with a precision of 80.2% and 73.4% and recall rates of 69.8% and 61.4%, for a total F-measure of 74.6% and 66.8%, respectively.

[2] `http://nhs.uk`
[3] `http://patientopinion.org.uk`
[4] `http://data.gov.uk/data`

Table 2: Comparative evaluation against WNA and SenticNet

Category	WNA	SenticNet	Sentic API
clinical service	59.12%	69.52%	78.06%
communication	66.81%	76.35%	80.12%
food	67.95%	83.61%	85.94%
parking	63.02%	75.09%	79.42%
staff	58.37%	67.90%	76.19%
timeliness	57.98%	66.00%	75.98%

As for the polarity detection, in turn, positivity and negativity of patient opinions were identified with particularly high precision (91.4% and 86.9%, respectively) and good recall rates (81.2% and 74.3%), for a total F-measure of 85.9% and 80.1%, respectively. More detailed comparative statistics are listed in Table 2, where the Sentic API is compared against WNA and SenticNet with respect to the polarity detection F-measures obtained on the 857 short stories.

6. CONCLUSION

Today user-generated contents are perfectly suitable for human consumption, but they remain hardly accessible to machines. Currently available information retrieval tools still have to face a lot of limitations. To bridge the conceptual and affective gap between word-level natural language data and the concept-level opinions and sentiments conveyed by them, we developed Sentic API, a common-sense based application programming interface that provides semantics and sentics associated with 15,000 natural language concepts.

We showed how Sentic API can easily be embedded in real-world NLP applications, specifically in the field of microblogging analysis, where statistical methods usually fail as syntax-based text processing works well only on formal-English documents and after training on big text corpora. We are keeping on developing the resource in a way that it can be continuously enhanced with more concepts from the always-growing Open Mind corpus and other publicly available common and common-sense knowledge bases. We are also developing novel techniques and tools to allow the Sentic API to be more easily merged with external domain-dependent knowledge bases, in order to improve the extraction of semantics and sentics from many different types of media and contexts.

7. REFERENCES

[1] C. Bazzanella. Emotions, language and context. In E. Weigand, editor, *Emotion in dialogic interaction. Advances in the complex*, pages 59–76. Benjamins, Amsterdam, 2004.

[2] C. Bosco, V. Patti, and A. Bolioli. Developing corpora for sentiment analysis and opinion mining: A survey and the Senti-TUT case study. *IEEE Intelligent Systems*, 28(2):55–63, 2013.

[3] M. Bradley and P. Lang. Affective norms for english words (ANEW): Stimuli, instruction manual and affective ratings. Technical report, The Center for Research in Psychophysiology, University of Florida., 1999.

[4] E. Cambria and A. Hussain. *Sentic Computing: Techniques, Tools, and Applications*. Springer, Dordrecht, Netherlands, 2012.

[5] E. Cambria, A. Hussain, C. Havasi, C. Eckl, and J. Munro. Towards crowd validation of the UK national health service. In *WebSci*, Raleigh, 2010.

[6] E. Cambria, A. Livingstone, and A. Hussain. The hourglass of emotions. In A. Esposito, A. Vinciarelli, R. Hoffmann, and V. Muller, editors, *Cognitive Behavioral Systems*, volume 7403 of *Lecture Notes in Computer Science*, pages 144–157. Springer, Berlin Heidelberg, 2012.

[7] E. Cambria, R. Speer, C. Havasi, and A. Hussain. SenticNet: A publicly available semantic resource for opinion mining. In *AAAI CSK*, pages 14–18, Arlington, 2010.

[8] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[9] A. Esuli and F. Sebastiani. SentiWordNet: A publicly available lexical resource for opinion mining. In *LREC*, 2006.

[10] L. García-Moya, H. Anaya-Sanchez, and R. Berlanga-Llavori. A language model approach for retrieving product features and opinions from customer reviews. *IEEE Intelligent Systems*, 28(3):19–27, 2013.

[11] M. Grassi. Developing HEO human emotions ontology. volume 5707 of *Lecture Notes in Computer Science*, pages 244–251. Springer, Berlin Heidelberg, 2009.

[12] C. Havasi, R. Speer, and J. Holmgren. Automated color selection using semantic knowledge. In *AAAI CSK*, Arlington, 2010.

[13] R. Horn and C. Johnson. Norms for vectors and matrices. In *Matrix Analysis*, chapter 5. Cambridge University Press, 1990.

[14] C. Hung and H.-K. Lin. Using objective words in SentiWordNet to improve sentiment classification for word of mouth. *IEEE Intelligent Systems*, 28(2):47–54, 2013.

[15] C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *Journal of Research of The National Bureau of Standards*, 45(4):255–282, 1950.

[16] M. Minsky. *The Emotion Machine: Commonsense Thinking, Artificial Intelligence, and the Future of the Human Mind*. Simon & Schuster, New York, 2006.

[17] H. Park and C. Jun. A simple and fast algorithm for k-medoids clustering. *Expert Systems with Applications*, 36(2):3336–3341, 2009.

[18] R. Plutchik. The nature of emotions. *American Scientist*, 89(4):344–350, 2001.

[19] S. Poria, A. Gelbukh, A. Hussain, D. Das, and S. Bandyopadhyay. Enhanced SenticNet with affective labels for concept-based opinion mining. *IEEE Intelligent Systems*, 28(2):31–30, 2013.

[20] R. Speer and C. Havasi. ConceptNet 5: A large semantic network for relational knowledge. In E. Hovy, M. Johnson, and G. Hirst, editors, *Theory and Applications of Natural Language Processing*, chapter 6. Springer, 2012.

[21] C. Strapparava and A. Valitutti. WordNet-Affect: An affective extension of WordNet. In *LREC*, pages 1083–1086, Lisbon, 2004.

[22] A. Tsai, R. Tsai, and J. Hsu. Building a concept-level sentiment dictionary based on commonsense knowledge. *IEEE Intelligent Systems*, 28(2):22–30, 2013.

[23] M. Wall, A. Rechtsteiner, and L. Rocha. Singular value decomposition and principal component analysis. In D. Berrar, W. Dubitzky, and M. Granzow, editors, *A Practical Approach to Microarray Data Analysis*, pages 91–109. Springer, 2003.

[24] A. Weichselbraun, S. Gindl, and A. Scharl. Extracting and grounding context-aware sentiment lexicons. *IEEE Intelligent Systems*, 28(2):39–46, 2013.

Section II:

Micropost Classification and Extraction

Evaluating Multi-label Classification of Incident-related Tweets

Axel Schulz*+ Eneldo Loza Mencía+ Thanh Tung Dang† Benedikt Schmidt*

*Telecooperation Lab
Technische Universität Darmstadt
Germany

†Knowledge Engineering Group
Technische Universität Darmstadt
Germany

+HCI Research
SAP AG, Darmstadt
Germany

{aschulz,benedikt.schmidt}@tk.informatik.tu-darmstadt.de eneldo@ke.tu-darmstadt.de thanh.tung.dang@sap.com

ABSTRACT

Microblogs are an important source of information in emergency management as lots of situational information is shared, both by citizens and official sources. It has been shown that incident-related information can be identified in the huge amount of available information using machine learning. Nevertheless, the currently used classification techniques only assign a single label to a micropost, resulting in a loss of important information that would be valuable for crisis management.

With this paper we contribute the first in-depth analysis of multi-label classification of incident-related tweets. We present an approach assigning multiple labels to these messages, providing additional information about the situation at-hand. An evaluation shows that multi-label classification is applicable for detecting multiple labels with an exact match of 84.35%. Thus, it is a valuable means for classifying incident-related tweets. Furthermore, we show that correlation between labels can be taken into account for these kinds of classification tasks.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering*; I.2.6 [**Artificial Intelligence**]: Learning

General Terms

Algorithms, Experimentation

Keywords

Microblogs, Multi-label Learning, Social Media

1. INTRODUCTION

Social media platforms are widely used by citizens for sharing information covering personal opinions about various topics (e.g., politics) as well as information about events such as incidents. In the latter case, citizens act as observers and create valuable incident-related information. For instance, during incidents such as the Oklahoma grass fires and the Red River floods in April 2009 [29], or the terrorist attacks on Mumbai [4], useful situational information was shared on Twitter. Also, Ushahidi, a social platform used for crowd-based filtering of information [15], was heavily used during the Haitian earthquake for labeling crisis-related information.

However, the discovery of incident-related information is a complex task, requiring the separation of valuable information from daily chatter in the vast amount of information created on social platforms. This can be realized based on techniques from data mining and machine learning. Classification is one method which can be utilized to extract relevant information from social networks (for tweets, see [23]). In a classification task, a system learns to label messages with exactly one label out of a predefined label set (e.g., "fire" or "crash"). This task is known as multi-class classification and widely used for text classification. However, during our research we found that assigning only one label would result in the loss of important situational information for decision making in crisis management. For instance, consider the following tweet:

> THIS CAR HIT THE FIRE HYDRANT AND CAUGHT FIRE....SOMEONE HOLIDAY ALTERED

A single label would necessarily lack relevant information. A better approach is the concurrent assignment of all three labels, which is known as multi-label learning. In the example, all labels ("fire", "crash", and "injuries") would be assigned concurrently using an appropriate learning algorithm. The example also shows that the assignment of multiple labels is not necessarily an independent process. Once the label for an incident type such as "crash" is assigned the probability of assigning the label "injuries" is changing. This dependency is known as label correlation and needs to be investigated in the context of multi-label learning.

With our analysis we want to investigate three important aspects of applying multi-label learning on incident-related tweets: (1) how to apply multi-label learners on tweets, (2) if the classification accuracy of multi-label classification approaches is comparable to the accuracy of multi-class classification approaches, and (3) if correlation between labels is a factor that needs to be taken into account for incident-related information. With this paper we contribute the first

Published as part of the #Microposts2014 Workshop proceedings, available online as CEUR Vol-1141 (http://ceur-ws.org/Vol-1141)

#Microposts2014, April 7th, 2014, Seoul, Korea.

in-depth analysis of multi-label classification of incident-related tweets. In summary, our contributions are twofold:

- We show that multi-label classification on incident-related tweets is applicable and able to detect the exact combinations of labels in 84.35% of the cases. Thus, we show that compared to common multi-class classification approaches, multi-label classification of incident-related tweets is a valuable means.
- We evaluate the influence of label correlation on the classification results of incident-related tweets. We show that for classification tasks label correlation needs to be taken into account.

The remainder of the paper is organized as follows. First, we describe and discuss related approaches. Second, the considered multi-label classification algorithms as well as the technical infrastructure (a machine learning pipeline) used for the analysis are presented. Next, we introduce our data collection setup and describe the evaluation of our approach. We close with a conclusion and future work.

2. RELATED WORK

Techniques of multi-label classification have been applied to domains such as text categorization [21, 13], music genre detection [20], or tag recommendation [7]. These application domains address long texts, images, or audio information. Text is probably one of the oldest domains in which the demand for categorization appeared, particularly multi-label categorization [25], with the first multilabel dataset (*Reuters-21578*) used in machine learning research being from the year 1987 [5, 8, 9]. Moreover, data is easily accessible and processable as well as vastly available. Hence, text classification was also one of the first research fields for multi-label classification and continues to be the most represented one among the commonly available benchmark datasets.[1]

A common application for texts is the classification of news articles [10, 18] for which the research focuses on scalability issues regarding the number of articles and especially the number of labels a text can be assigned to, which can sometimes go up to the thousands [11, 26]. News texts, as well as abstracts from scientific papers [14] or radiology reports [16] may sometimes be relatively short, but they are usually still structured and homogeneous. This kind of multi-label text classification problems were very well analyzed in the past and the used approaches showed to be effective (we refer the interested reader to the cited recent works).

In contrast, texts such as tweets are mostly unstructured and noisy, because of their limitations in size and the often used colloquial language. Related work on such short texts with a focus on solving multi-class problems exists, e.g., for sentiment analysis [24] or incident detection and classification [23]. In contrast to these approaches, this paper focuses on the use of multi-label classification for tweets.

Applying multi-label learning on very short texts is a topic of open research. Only two respective examples are known to the authors: Sajnani et al. [19] and Daxenberger et al. [1]. Sajnani et al. provided a preliminary analysis of multi-label classification of Wikipedia barnstar texts. Barnstars can be awarded by Wikipedia authors and contain a short textual explanation why they have been awarded. In this case, labels for seven work domains have to be differentiated. The authors show which features can be extracted from short texts for multi-label classification and evaluate several multi-label classification approaches. Daxenberger et al. categorize individual edits into non-exclusive classes like *vandalism*, *paraphrase*, etc.

Summarized, although many related approaches cope with multi-class classification of short texts such as microblogs, multi-label classification is an open research issue. Especially for the domain of crisis management, no prior research on this topic exists.

[1]Cf. `http://mulan.sourceforge.net/datasets.html` [28] and `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multilabel.html` repositories.

3. MULTI-LABEL CLASSIFICATION

In this section, we give an overview on multi-label classification. Multi-label classification refers to the task of learning a function that maps instances $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,a}) \in \mathcal{X} \subseteq \mathbb{R}^a$ to label subsets or label vectors $\mathbf{y}_i = (y_{i,1}, \ldots, y_{i,n}) \in \{0,1\}^n$, where $\mathcal{L} = \{\lambda_1, \ldots, \lambda_n\}$, $n = |\mathcal{L}|$ is a finite set of predefined labels and where each label attribute y_i corresponds to the absence (0) or presence (1) of label λ_i. Thus, in contrast to multi-class classification, alternatives are not assumed to be mutually exclusive, such that multiple labels may be associated with a single instance.

This makes multi-label data particularly interesting from the learning perspective, since, in contrast to binary or multi-class classification, there are label dependencies and interconnections in the data which can be detected and exploited in order to obtain additional useful information or just better classification performance. Some examples for our particular Twitter dataset were already shown up in the introduction. As we show, around 15% of our tweets could be assigned to more than one label, thus, we believe that it is not unusual to encounter tweets with several possible labels, so that in our opinion the view of microblogs as multi-labeled data seems more natural, more realistic, and more general. Nonetheless, previous work focuses on the multi-class labeling of tweets and this is the first work known to the authors which tries to exploit label dependencies on tweets.

In the following, we will describe commonly used approaches for multi-label classification: Binary Relevance (BR), Label Powerset (LP), and Classifier Chains (CC). All described techniques are based on the decomposition or transformation of the original multi-label problem into single-label binary problems, as most multi-label techniques do [27]. An illustration of these techniques is presented in Figure 1. This has the advantage that we can use state-of-the-art text classification algorithms for learning the binary problems such as support vector machines [25, 6]. We will also have a closer look at each classification approach with respect to taking dependencies between labels into account. Two of the used approaches are specifically tailored in order to cope with such dependencies.

3.1 Binary Relevance

The most common approach for multi-label classification is to use an ensemble of binary classifiers, where each classifier predicts if an instance belongs to one specific class or not. The union of all classes that were predicted is taken

$\mathbf{x}_i$	Labels $\in \{0,1\}^n$
$\mathbf{x}_1$	$(y_{1,1}, \ldots, y_{1,n})$
$\mathbf{x}_2$	$(y_{2,1}, \ldots, y_{2,n})$
$\vdots$	$\vdots \quad \ddots \quad \vdots$

(a) input training set

$\mathbf{x}_i$	Class $\in \{1, \ldots, 2^n\}$
$\mathbf{x}_1$	$\sigma(\mathbf{y}_1)$
$\mathbf{x}_2$	$\sigma(\mathbf{y}_2)$
$\vdots$	$\vdots$

(b) *label powerset* (LP) decomposition

$\mathbf{x}_i$	Class$_1 \in \{0,1\}$
$\mathbf{x}_1$	$y_{1,1}$
$\mathbf{x}_2$	$y_{2,1}$
$\vdots$	$\vdots$

$\cdots$

$\mathbf{x}_i$	Class$_n \in \{0,1\}$
$\mathbf{x}_1$	$y_{1,n}$
$\mathbf{x}_2$	$y_{2,n}$
$\vdots$	$\vdots$

(c) *binary relevance* (BR) decomposition

$\mathbf{x}'_i$	Class$_1 \in \{0,1\}$
$\mathbf{x}_1$	$y_{1,1}$
$\mathbf{x}_2$	$y_{2,1}$
$\vdots$	$\vdots$

$\cdots$

$\mathbf{x}'_i \in \mathbb{R}^a \times \{0,1\}^{n-1}$	Class$_n \in \{0,1\}$
$(\mathbf{x}_1, y_{1,1}, \ldots, y_{1,n-1})$	$y_{1,n}$
$(\mathbf{x}_2, y_{2,1}, \ldots, y_{2,n-1})$	$y_{2,n}$
$\vdots$	$\vdots$

(d) *classifier chains* (CC) decomposition

Figure 1: Decomposition of multi-label training sets into multiclass (LP) or binary (BR, CC) problems. $\mathbf{x}'_i$ denotes the augmented instance. During prediction, $y_{i,1}, y_{i,2}, \ldots$ in the extended input space is replaced by the predictions by $h_1^{CC}, h_2^{CC}, \ldots$ (see text).

as the multi-label output. This approach is comparable to classical one-against-all for a multi-class problem. Formally, we convert a training example pair $(\mathbf{x}_i, \mathbf{y}_i)$ into n separate pairs $(\mathbf{x}_i, y_{i,j})$, $j = 1 \ldots n$, one for each of the n base classifiers h_j. The predicted labels $\hat{y}_j$ for a test instance $\mathbf{x}$ are then the result of $h_j(\mathbf{x}) \in \{0,1\}$.

This method is fast and simple, however, it is not able to take label dependencies into account since each base classifier is trained independently from the other classifiers. As was recently stated by Dembczynski et. al [2], this is not necessarily a disadvantage if the objective is to obtain good label-wise predictions, such as measured by the Hamming loss (cf. Section 5). Therefore, BR serves as a fairly good performing baseline for our experiments.

3.2 Label Powerset

The basic idea of this algorithm is to transform multi-label problems into a multi-class classification problem by considering each member of the powerset of labels in the training set as a single class. Hence, each training example is converted into $(\mathbf{x}_i, \sigma(\mathbf{y}_i))$ with σ, σ^{-1} denoting a bijective function that maps between the label powerset of $\mathcal{L}$ and a set of 2^n meta-classes. The classifier h^{LP} is trained e.g. with one-against-all (like in our setting), and the prediction for $\mathbf{x}$ is obtained with $\sigma^{-1}(h^{LP}(\mathbf{x}))$.

LP takes label dependencies into account to some extent, as each distinct occurrence of a label pattern is treated as a new class. It is hence able to model the joint label distribution, but not explicitly and directly specific dependencies (correlations, implications, etc.) between labels. As a consequence, LP is tailored towards predicting exactly the correct label combination. As it is pointed out in [2] and contrary to what one may believe at first, this stays usually in contrast to predicting correctly each label individually (BR), i.e. we usually have a trade-off between both objectives.

In addition to the obvious computational costs problem due to the exponential grow of meta-labels, the sparsity of some label combinations, especially with an increasing number of labels, often causes that some classes contain only few examples. This effect can also be observed in our data, cf. Table 2.

3.3 Classifier Chains

As stated before in Section 1, it is very likely in our dataset that injured people are mentioned when also any incident type is mentioned (200 of 967 cases). On the other hand, it seems almost a matter of course that there was an incident if there is an injured person. Although this only happens in 200 out of 232 cases in our data we consider it relevant for larger data sets. The classifier chains approach (CC) of Read et al. [17] is able to directly capture such dependencies and has therefore become very popular recently.

The idea of this approach is to construct a chain of n binary classifiers h_j^{CC}, for which (in contrast to BR) each binary base classifier h_j^{CC} depends on the predictions of the previous classifiers $h_1^{CC} \ldots h_{j-1}^{CC}$. Particularly, we extent the feature space of the training instances for the base classifier h_j^{CC} to $((x_{i,1} \ldots x_{i,a}, y_{i,1} \ldots y_{i,j-1}), y_{i,j})$. Since the true labels y_i are not known during prediction, CC uses the predictions of the preceding base classifiers instead. Hence, the unknown y_j are replaced by the predictions $\hat{y}_j = h_j^{CC}(\mathbf{x}, \hat{y}_1 \ldots \hat{y}_{j-1})$.

This shows up one problematic aspect of this approach, namely the order of the classifiers in the chain. Depending on the ordering, CC can only capture one direction of dependency between two labels. More specifically, CC can only capture the dependencies of y_i on $y_1, \ldots, y_{i-1}$, but there is no possibility to consider dependencies of y_i on $y_{i+1}, \ldots, y_n$. Recovering our example from the beginning, we can either learn the dependency of the label *incident* given *injury* or the other way around, but not both. In addition, the effect of error propagation caused by the chaining structure may also depend on the label permutation. We will evaluate the effect of choosing different orderings for our particular dataset later on in Section 5.3.

Furthermore, CC has advantages compared to LP. CC is considered to predict the correct label-set, such as LP [2], but unlike LP, CC is able to predict label combinations which were not seen beforehand in the training data. In addition, the imbalance between positive and negative training examples is generally lower than for LP.

4. MULTI-LABEL CLASSIFICATION OF INCIDENT-RELATED TWEETS

In the following, the data used for multi-label classification of incident-related tweets is described in detail. The taken approach is composed of three steps. As a first step, unstructured text has to be converted into structured text. As a second step, the structured information needs to be transformed to features that can be used by a multi-label learner. Third, these features are used to train and evaluate a classifier.

4.1 Preprocessing of Unstructured Text

Our overall goal is to apply text mining on short docu-

ments that are present in social media, thus, they need to be represented by a set of features. As texts in social media are mostly unstructured, they first need to be converted into a representation which enables feature generation. Hence, as a first step, we apply Natural Language Processing. Firstly, we remove all re-tweets as these are just duplicates of other tweets and do not provide additional information. Secondly, @-mentions of Twitter users are removed from the tweet message as we want to prevent overfitting towards certain user tokens. Before further processing is applied, the text is converted to Unicode, as some tweets contain non-Unicode characters. Third, abbreviations are resolved using a dictionary of abbreviations based on the data provided by the Internet Slang Dictionary&Translator[2]. Then, we identify and replace URLs with a common token "URL". As a next step, stopwords are removed. This is important as very frequent words have limited influence when it comes to classifying tweets due to their relative frequency. Based on the resulting text, we conduct tokenization. Thus, the text is divided into discrete words (tokens) based on different delimiters such as white spaces. Every token is then analyzed and non-alphanumeric characters are removed or replaced. Also, lemmatization is applied to normalize all tokens. Additionally to the common NLP processing steps, we identify and replace location mentions such as "Seattle" with a common token to allow semantic abstraction. For this, we use the approach presented in [23] to detect named entities referring to locations (so-called location mentions) in tweets and to replace them with two tokens "LOC" and "PLACE".

4.2 Feature Generation

After finishing the initial preprocessing steps, we extracted several features from the tweets that are used for training a classifier. We conducted a comprehensive feature selection, analyzing the value of each feature for the overall classification performance. We compared word-n-grams, char-n-grams, TF-IDF [12] scores as well as syntactic features such as the number of explanation marks, question marks, and upper case characters. We found that the following features are the most beneficial for our classification problems:

- Word 3-gram extraction: We extract word three-grams from the tweet message. Each 3-gram is represented by two attributes. One attribute indicating the presence of the 3-gram and another attribute indicating the frequency of the 3-gram.
- Sum of TF-IDF scores: For every document we calculate the accumulated TF-IDF (term-frequency inverse-document-frequency) score [12] based on the single TF-IDF scores of each term in the document. The rational behind this is to create a similarity score which is not as strict as traditional TF-IDF scores, but allows forming of clusters of similar documents.
- Syntactic features: Along with the features directly extracted from a tweet, several syntactic features are expected to improve the performance of our approach. People might tend to use a lot of punctuations, such as explanation marks and question marks, or a lot of capitalized letters when they are reporting some incident. In this case, we extract the following features: the number of '!' and '?' in a tweet and the number of capitalized characters.
- Spatial features: As location mentions are replaced with a corresponding token, they appear as word unigrams in our model and can therefore be regarded as additional features.

4.3 Dataset

We focus on three different incident types throughout the paper in order to differentiate incident-related tweets. Three classes have been chosen, because we identified them as the most common incident types using the Seattle Real Time Fire Calls dataset[3], which is a frequently updated source for official incident information. We included also *injury* as an additional label. This results in four labels consisting of very common and distinct incident types and the injury label: Fire, Shooting, Crash, and Injury.

We collected public tweets in English language using the Twitter Search API, which provides geotagged tweets as well as tweets for which Twitter inferred a geolocation based on the user profile. For the collection, we used a 15km radius around the city centers of Seattle, WA and Memphis, TN. We focused on only two cities, as for our analyses we are interested in the stream of tweets for these cities and a specific time period instead of a scattered sample of the world, which could be retrieved using the Twitter Streaming API. This gave us a set of 7.5M tweets collected from 11/19/12 to 02/07/13. Though we know about the limitations of the Search API, we think that we collected a relevant sample for our experiments.

The dataset was further reduced to be usable for high quality labeling as well as the machine learning experiment. We first identified and extracted tweets mentioning incident-related keywords. Compared to other approaches that completely rely on filtering using hashtags, we take the whole message into account for identifying incident-related keywords. We retrieved a set of different incident types using the "Seattle Real Time Fire 911 Calls" dataset and defined one general keyword set with keywords that are used in all types of incidents like 'incident', 'injury', 'police', etc. For each incident type, we further identified specific keywords. For instance, for the incident type 'Motor Vehicle Accident Freeway' we use the keywords 'vehicle', 'accident', and 'road'. Based on these words, we use WordNet[4] to extend this set by adding the direct hyponyms. For instance, the keyword 'accident' was extended with 'collision', 'crash', 'wreck', 'injury', 'fatal accident', and 'casualty'. Based on these incident-related keywords, we filtered the datasets. Furthermore, we removed all re-tweets, as the originated tweets are also contained in our datasets and only these are needed for our experiments. Based on this filtered dataset, we randomly selected 20.000 tweets.

The selected tweets have been labeled manually by one researcher of our department. Out of these tweets, we randomly selected 2.000 tweets for further re-labeling for our multi-label classification problem. Those tweets were manually examined by five researchers using an online survey. To assign the final coding, we differentiated between two types of agreement:

[2] http://www.noslang.com

[3] http://data.seattle.gov

[4] http://wordnet.princeton.edu

Table 1: Overview of real-world incident types used for extraction of incident-related keywords as well as and the number of extracted keywords for keyword-based classification approach.

Class	**Fire**	**Shooting**	**Crash**	**Injury**
Real-World Incident Type	Fire In Building	Assault w/Weap	Motor Vehicle Accident	-
	Fire In Single Family Res	Assault w/Weapons Aid	Motor Vehicle Accident Freeway	
	Automatic Fire Alarm Resd		Medic Response Freeway	
	Auto Fire Alarm		Car Fire	
			Car Fire Freeway	
# of Keywords	148	36	73	23

Table 2: Distribution of the 10 label combinations occurring in the 2000 tweets of the dataset.

Label Combination	Number of Tweets
{}	971
{Fire}	313
{Shooting}	184
{Crash}	268
{Injury}	32
{Crash, Fire}	2
{Injury, Crash}	47
{Injury, Shooting}	149
{Injury, Fire}	33
{Injury, Fire, Crash}	1

- if four out of five coders agree on one label, only this label is assigned
- if less than four coders agree on one label, all labels which at least two coders assumed as correct are assigned as possible labels and further verified in a group discussion

The final labeled dataset consists of 10 different label combinations. The distribution for every combination is outlined in Table 2. The distribution indicates that around 15% (232) of all tweets in our dataset have been labeled with multiple labels. Another observation is that almost exactly 50% of the tweets do not have any label assigned, which is rather unusual compared to typically used and analyzed multi-label datasets[5]. In addition, the label cardinality, i.e., the average number of labels assigned to an instance, is around 0.59, whereas common datasets have at least more than 1 assigned. On the other hand, this is mainly due to the low number of total labels, since the label density (the average percentage of labels which are true) is 15%, which is a relatively high value. From a multi-label learning perspective, this is an interesting property of this dataset since it is not clear how commonly used techniques will behave under this circumstance. For example, many algorithms ignore instances without any label given.

[5]We refer to the repository at `http://mulan.sourceforge.net/datasets.html` for an overview of the statistics of the commonly used benchmark datasets in multi-label classification

5. EVALUATION

In the following section, we provide the evaluation results for the presented multi-label classification approaches on our dataset. We also present the result for a keyword-based approach as a simple way for conducting multi-label classification.

5.1 Evaluation Setup

We performed our experiments with Mulan, an open-source library for multi-label classification based on Weka [28]. We used two learners for our evaluation. First, we use the LibLinear implementation of support vector machines with linear kernel [3] as our base learner. We use the default settings, as we found that additional parameter optimization was not beneficial for improving the overall classification results. Second, we used the Weka implementation of Naive Bayes. The results were obtained using 10-fold cross validation.

The evaluation of multi-label problems requires different measures compared to those used for multi-class problems. In our paper, we use the following metrics:

Exact Match: Exact match is the percentage of the m test instances for which the labelsets were exactly correctly classified (with $[[z]]$ as indicator function returning 1 if z is true, otherwise 0)

$$ExactMatch(h) = \frac{1}{m}\sum_{i=1}^{m}[[y_i = h(\mathbf{x}_i)]] \quad (1)$$

Hamming Loss: The instance-wise Hamming loss [22] is defined as the percentage of wrong or missed labels compared to the total number of labels in the dataset. In this case, it is taken into account that an incorrect label is predicted and that a relevant label is not predicted. As this is a loss function, the optimal value is zero.

Recall, Precision and F1: We use micro-averaged precision and recall measures to evaluate our results, i.e., we compute a two-class confusion matrix for each label ($y_i = 1$ vs. $y_i = 0$) and eventually aggregate the results by (component-wise) summing up all n matrices into one global confusion matrix (cf. [27]). Recall and precision is computed based on this global matrix in the usual way, F1 denotes the unweighted harmonic mean between precision and recall. In Section 5, we also report recall, precision and F1 for each label using the label-wise confusion matrices.

5.2 Results for Keyword-Based Filtering

As mentioned before, we use a keyword-based pre-filtering for selecting an initial set of tweets that is suitable for la-

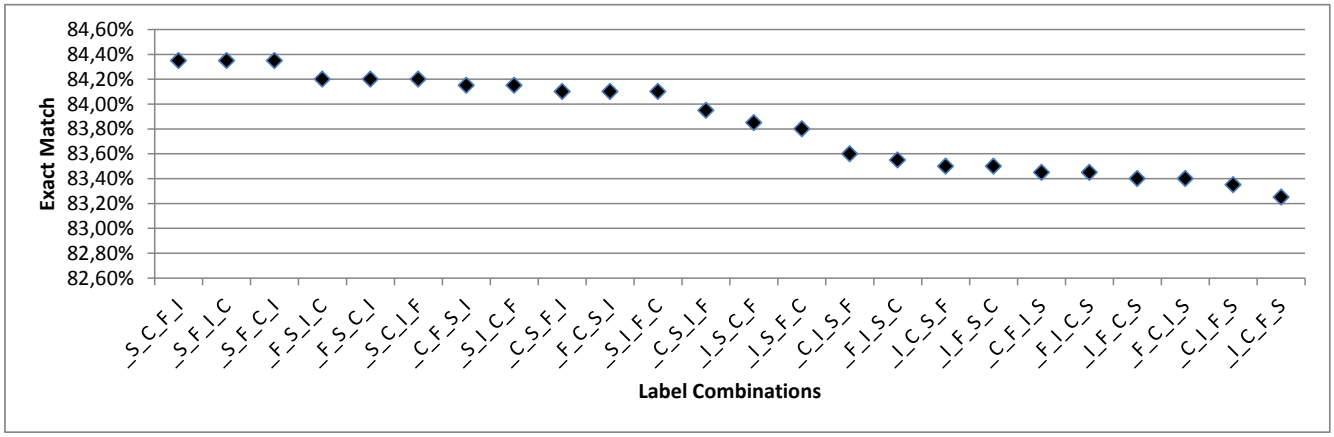


Figure 2: Percentages of exact matches for all label combinations.

beling. A first and simple approach for detecting incident related tweets is to use these keywords for classification.

In Table 1, the real-world incident types from the Seattle Real Time Fire Calls dataset and the corresponding number of extracted keywords is shown. For the injury class, no specific type in the Seattle dataset could be found, thus, we extended the set with a manually created list of keywords and their direct hyponyms.

The results for classifying each individual class are shown in Table 3. The results indicate that precision as well as recall are rather low. Only for the fire class a high recall could be achieved.

Table 3: Precision and recall for each individual label when applying keyword-based classification.

	Shooting	Fire	Crash	Injury
Precision	31.59%	54.12%	15.04%	63.64%
Recall	68.77%	95.99%	49.37%	37.40%

Furthermore, if the keywords would be used for applying multi-label classification, a precision of 32.22% and a recall of 64.90% is achieved, which is a rather bad result. Also exact match (28.45%) and h-loss (27.08%) are bad, thus, we conclude that with simple keyword-based filtering, multi-label classification cannot be done accurately.

5.3 Results for Multi-Label Classification

As a first step, we coped with the question if correlation between labels is taken into account and beneficial for the classification results. Thus, we evaluated all different label sequences using the classifier chains algorithm for our labels Fire (F), Shooting (S), Crash (C), and Injury (I). The values for exact match for each sequence are shown in Figure 2 (using SVM as our base learner).

The results indicate that the label sequence has indeed an influence on the classification performance. In our case, we get a difference of 1% between the best sequence Shooting, Crash, Fire, Injury and the worst Injury, Crash, Fire, Shooting. Also, we see that the Injury label is best used after incident labels have been classified - for the best cases even as one of the last labels in the sequence. It is also remarkable that classifying Shooting as first label followed up by either Crash or Fire is always a good option. This can be explained on the one hand by the generally good individual prediction performance for Shooting (cf., Table 5), hence leading to low error propagation, and on the other hand by the resulting label dependencies given the Shooting label is known: for instance, we can see from Table 2 that we can safely exclude Crash or Fire if there was a Shooting. This shows that our initial assumption that correlation between labels needs to be taken into account is true.

Based on the respective best (MAX) and the worst sequence (MIN), we compared CC to the multi-label approaches with the two different base learners. In Table 4 these evaluation results are shown. The first observation is that Naive Bayes is not adequate for classifying tweets, since though it achieves the best recall values using CC, this is in exchange of very low results on the remaining metrics and approaches. We will therefore focus on the results obtained by applying LibLinear as base learner. The results show that, if there is the opportunity of pre-optimizing the ordering of the labels, e.g., by performing a cross-validation on the training data, then classifier chains is able to slightly outperform the other approaches, which is most likely because the label correlation is valuable. This is also reflected in the good performance with respect to exact match, where the worst CC even outperforms LP, which is particularly tailored towards matching the exact label combination. Note also that LP is a common approach used for circumventing the need for a multi-label classification by creating meta-classes, as already mentioned in the introduction. However, this approach is always inferior to the compared approaches, which demonstrates the need for more advanced techniques in this particular use case.

We can also observe that improving the prediction of the exact label combinations may come at the expense of reducing the performance on label-wise measures, since the additional features used by CC generally lead to a higher potential deterioration (MIN) than potential improvement (MAX) for Hamming loss, recall, precision and F1, whereas for exact match this is not as clear.

As a last evaluation step, we evaluated the accuracy of each approach for every individual label. This is important as we want to understand how well a classifier performs for each label. The following Table 5 depicts the accuracy of individual labels using SVM with the best label order.

Table 4: Results for the different multi-label approaches and base learners obtained by cross-validation.

	Naive Bayes				SVM			
	BR	LP	CC - MIN	CC - MAX	BR	LP	CC - MIN	CC - MAX
Exact Match	59.60%	66.95%	71.15%	**72.45%**	83.85%	83.05%	83.25%	**84.35%**
H-Loss	15.02%	14.08%	9.400%	**9.175%**	4.688%	5.313%	4.900%	**4.588%**
F1	52.19%	55.37%	72.90%	**73.61%**	83.55%	81.53%	82.80%	**84.02%**
Precision	52.40%	55.34%	66.84%	**67.92%**	**93.61%**	90.28%	92.75%	93.46%
Recall	51.98%	55.39%	79.63%	**80.35%**	75.44%	74.35%	74.72%	**76.47%**

Table 5: Precision and recall for each individual label.

	BR (SVM)		LP (SVM)		CC (SVM)	
	Prec.	Recall	Prec.	Recall	Prec.	Recall
Shooting	95.7%	79.3%	92.0%	76.9%	95.7%	79.3%
Fire	94.7%	82.0%	90.3%	83.0%	93.3%	83.7%
Crash	90.8%	77.4%	88.0%	78.3%	90.9%	78.3%
Injury	92.9%	59.5%	91.1%	54.6%	93.0%	61.0%

The results show that the precision for individual labels is high with about 90% to 95% for each label, which is much better compared to the keyword-based classification. The differences between all approaches are nearly the same, thus, all approaches seem to be appropriate for classifying the individual labels. However, the recall drops significantly, depending on the label type. For instance, injuries often remain undetected. In this case, classifier chains show the best results for precision and recall. Note that the results for BR and CC on Shooting are the same, since the first classifier in the CC ordering is exactly trained like the corresponding BR classifier (cf. also Figure 1). This also shows that along the chain, CC slightly reduces the good precision of BR in exchange of improved recall.

5.4 Discussion

Though the results show the advantage of multi-label classification, we want to understand the limitations of our approach. Thus, we first created a confusion matrix for the classifier chains approach with the best label order. The matrix shows that most misclassifications occur due to an assignment of instances to the "no incident" label combination {}. The other wrong classifications are mostly a result of not detecting the injury label or of predicting it wrongly.

Table 6: Confusion matrix. The rows indicate the predicted/true label combinations and the columns the true/predicted ones.

	∅	F	C	F,C	I	F,I	C,I	F,C,I	S	F,S	I,S
∅	924	16	24	0	0	0	0	0	3	0	4
F	49	261	0	0	0	3	0	0	0	0	0
C	54	0	213	1	0	0	0	0	0	0	0
F,C	1	1	0	0	0	0	0	0	0	0	0
I	16	0	1	0	11	0	0	0	1	0	3
F,I	5	10	0	0	1	17	0	0	0	0	0
C,I	8	0	12	0	3	0	23	0	0	0	1
F,C,I	1	0	0	0	0	0	0	0	0	0	0
S	33	4	0	0	1	0	0	0	142	0	4
F,S	0	0	0	0	0	0	0	0	0	0	0
I,S	26	0	0	0	5	0	0	0	22	0	96

The following misclassified tweets show examples for such wrongly classified instances:

> "Tacoma Fire Department replaces 3 fire engines with pickup trucks: TACOMA Cutbacks within the Tacoma Fire... http://t.co/jPe2kuKG" ({} -> {F})

> "This girl is on fire. This girl is on fire. She's walking on fire. This girl is on fire - Alicia Keys #deep", ({} -> {S})

> "NeoMemphis News: Massive fire at factory in Ripley: Action News 5 is on the scene of a factory fire at ... http://t.co/brfnVbWp #memphis", ({F} -> {F,I})

The examples show that certain words such as "fire" or digits in the message might lead to wrong classifications. This could be avoided by adding additional features or with a larger training set.

In this section we have first shown that a simple keyword-based classification approach is not suitable for multi-label classification. Second, we presented results of state-of-the-art multi-label classification approaches and we showed that these perform quite well for classifying incident-related tweets. Compared to current approaches for the classification of microblogs, which rely on assigning only one label to an instance, the results show that it is possible to infer important situational information with only *one* classification step. The results also indicate that the label sequence has an influence on the classification performance, thus, this factor should be taken into account for following approaches.

6. CONCLUSION

In this paper we have shown how to apply multi-label learning on social media data for classification of incident-related tweets. Furthermore, we analyzed that we are able to identify multiple labels with an exact match of 84.35%. This is an important finding, as multiple labels assigned with one classification approach provide important information about the situation at-hand, which could not be easily derived from previously used multi-class classification approaches. Furthermore, we have shown that the natural relation of labels, which represents for instance the relation between incidents and injuries in the real-world, can be used and exploited by classification approaches in order to obtain better results.

For future work, we aim to add costs to our classifications. For instance, not detecting incident labels should be heavily punished compared to misclassifying the incident type. Furthermore, we aim to improve the overall performance of our approach by taking different features and a larger training set into account.

Acknowledgements

This work has been partly funded by the German Federal Ministry for Education and Research (BMBF, 01|S12054).

References

[1] J. Daxenberger and I. Gurevych. A corpus-based study of edit categories in featured and non-featured wikipedia articles. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pages 711–726, Dec. 2012.

[2] K. Dembczynski, W. Waegeman, W. Cheng, and E. Hüllermeier. On label dependence and loss minimization in multi-label classification. *Machine Learning*, 88(1-2):5–45, 2012.

[3] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, aug 2008.

[4] R. Goolsby. Lifting Elephants: Twitter and Blogging in Global Perspective. In *Social Computing and Behavioral Modeling*. 2009.

[5] P. J. Hayes and S. P. Weinstein. CONSTRUE/TIS: A system for content-based indexing of a database of news stories. In A. T. Rappaport and R. G. Smith, editors, *Proceedings of the 2nd Conference on Innovative Applications of Artificial Intelligence (IAAI-90), May 1-3, 1990, Washington, DC, USA*, IAAI '90, pages 49–64. AAAI Press, Chicago, IL, USA, 1991.

[6] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In C. Nédellec and C. Rouveirol, editors, *Proceedings of 10th European Conference on Machine Learning (ECML-98)*, pages 137–142, Chemnitz, Germany, 1998. Springer-Verlag.

[7] I. Katakis, G. Tsoumakas, and I. P. Vlahavas. Multilabel text classification for automated tag suggestion. In *Proceedings of the ECML/PKDD-08 Workshop on Discovery Challenge*, Antwerp, Belgium, 2008.

[8] D. D. Lewis. An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Devlopment in Information Retrieval*, pages 37–50, 1992.

[9] D. D. Lewis. Reuters-21578 text categorization test collection distribution 1.0. README file (V 1.3), May 2004.

[10] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. RCV1: A New Benchmark Collection for Text Categorization Research. *Journal of Machine Learning Research*, 5:361–397, 2004.

[11] E. Loza Mencía and J. Fürnkranz. Efficient pairwise multilabel classification for large-scale problems in the legal domain. In *Proc. ECML-PKDD-2008*, volume 5212 of *LNCS*, pages 50–65, Antwerp, Belgium, 2008. Springer.

[12] C. D. Manning, P. Raghavan, and H. Schütze. *An Introduction to Information Retrieval*, pages 117–120. Cambridge University Press, 2009.

[13] A. McCallum. Multi-label text classification with a mixture model trained by EM. In *AAAI'99 Workshop on Text Learning*, pages 1–7, 1999.

[14] A. Montejo Ráez, L. A. Ureña López, and R. Steinberger. Adaptive selection of base classifiers in one-against-all learning for large multi-labeled collections. In *Advances in Natural Language Processing, 4th International Conference (EsTAL 2004), Alicante, Spain, October 20-22, Proceedings*, volume 3230 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2004.

[15] O. Okolloh. Ushahidi, or 'testimony': Web 2.0 tools for crowdsourcing crisis information. *Participatory Learning and Action*, 59(January):65–70, 2008.

[16] J. P. Pestian, C. Brew, P. Matykiewicz, D. Hovermale, N. Johnson, K. B. Cohen, and W. Duch. A shared task involving multi-label classification of clinical free text. In *Proceedings of the Workshop on BioNLP 2007: Biological, Translational, and Clinical Language Processing*, pages 97–104. Association of Computational Linguistics, June 2007.

[17] J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. *Machine Learning*, 85(3):333–359, June 2011.

[18] T. N. Rubin, A. Chambers, P. Smyth, and M. Steyvers. Statistical topic models for multi-label document classification. *Machine Learning*, 88(1-2):157–208, 2012.

[19] H. Sajnani, S. Javanmardi, D. W. McDonald, and C. V. Lopes. Multi-label classification of short text: A study on wikipedia barnstars. In *Analyzing Microtext*, 2011.

[20] C. Sanden and J. Z. Zhang. Enhancing multi-label music genre classification through ensemble techniques. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 705–714. ACM, 2011.

[21] R. E. Schapire and Y. Singer. Boostexter: A boosting-based system for text categorization. *Machine learning*, 39(2-3):135–168, 2000.

[22] R. E. Schapire and Y. Singer. BoosTexter: A Boosting-based System for Text Categorization. *Machine Learning*, 39(2/3):135–168, 2000.

[23] A. Schulz, P. Ristoski, and H. Paulheim. I see a car crash: Real-time detection of small scale incidents in microblogs. In P. Cimiano, M. Fernàndez, V. Lopez, S. Schlobach, and J. Völker, editors, *The Semantic Web: ESWC 2013 Satellite Events*, number 7955 in Lecture Notes in Computer Science, pages 22–33. Springer Berlin Heidelberg, 2013.

[24] A. Schulz, T. D. Thanh, H. Paulheim, and I. Schweizer. A fine-grained sentiment analysis approach for detecting crisis related microposts. In *Proceedings of the 10th International Conference on Information Systems for Crisis Response and Management (ISCRAM)*, pages 846 – 851, May 2013.

[25] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, Mar. 2002.

[26] G. Tsoumakas, I. Katakis, and I. P. Vlahavas. Effective and efficient multilabel classification in domains with large number of labels. In *Proceedings ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08)*, 2008.

[27] G. Tsoumakas, I. Katakis, and I. P. Vlahavas. Mining multi-label data. In O. Maimon and L. Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, pages 667–685. Springer, 2010.

[28] G. Tsoumakas, E. Spyromitros Xioufis, J. Vilcek, and I. P. Vlahavas. Mulan: A java library for multi-label learning. *Journal of Machine Learning Research*, 12:2411–2414, 2011. Software available at http://mulan.sourceforge.net/.

[29] S. Vieweg, A. L. Hughes, K. Starbird, and L. Palen. Microblogging during two natural hazards events: what twitter may contribute to situational awareness. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems Pages (CHI'10)*, 2010.

Combining Named Entity Recognition Methods for Concept Extraction in Microposts

Štefan Dlugolinský
upsysdlu@savba.sk

Peter Krammer
upsypkra@savba.sk

Marek Ciglan
upsymaci@savba.sk

Michal Laclavík
laclavik.ui@savba.sk

Ladislav Hluchý
upsylhlu@savba.sk

Institute of Informatics, Slovak Academy of Sciences
Dúbravská cesta 9
845 07 Bratislava, Slovakia

ABSTRACT

NER in microposts is a key and challenging task of mining semantics from social media. Our evaluation of a number of popular NE recognizers over a micropost dataset has shown a significant drop-off in results quality. Current state-of-the-art NER methods perform much better on formal text than on microposts. However, the experiment provided us with an interesting observation – although individual NER tools did not perform very well on micropost data, we have received recall over 90% when we merged all the results of the examined tools. This means that if we would be able to combine different NE recognizers in a meaningful way, we might be able to get NER in microposts of an acceptable quality. In this paper, we propose a method for NER in microposts, which is designed to combine annotations yielded by existing NER tools in order to produce more precise results than input tools alone. We combine NE recognizers utilizing ML techniques, namely decision tree and random forest using the C4.5 algorithm. The main advantage of the proposed method lies in the possibility of combining arbitrary NER methods and in its application on short, informal texts. The evaluation on a standard dataset shows that the proposed approach outperforms underlying NER methods as well as a baseline recognizer, which is a simple combination of the best underlying recognizers for each target NE class. To the best of our knowledge, up-to-date, the proposed approach achieves the highest F_1 score on the #MSM2013 dataset.

Categories and Subject Descriptors

1.2.7 [**Natural language processing**]: Language parsing and understanding, Text analysis.

Keywords

named entity recognition, machine learning, microposts

Published as part of the #Microposts2014 Workshop proceedings, available online as CEUR Vol-1141 (http://ceur-ws.org/Vol-1141)

#Microposts2014, April 7th, 2014, Seoul, Korea.

1. INTRODUCTION

A significant growth of social media interaction can be observed in recent years. People are able to interact through the Internet from almost anywhere at anytime. They can share their experience, thoughts and knowledge instantly and they do it in mass dimensions. The easiest and probably the most popular way of interaction on the Web is through microposts – short text messages posted on the Web. There is a plenty of services offering such communication, notorious examples of microposts include tweets, Facebook statuses, comments, Google+ posts, Instagram photos. Microposts analysis has a big potential in hidden knowledge that can be used in wide range of domains like emergency response, public opinion assessment, business or political sentiment analysis and many more. The most important task in order to analyze and make sense of microposts is the Named Entity Recognition (NER). NER in microposts is a challenging problem because of a limited size of a single micropost, prevalence of term ambiguity, noisy content, multilingualism [2]. These are the main reasons why existing NER methods perform better on formal newswire text than on microposts and there is clearly a space for new methods of NER designed for social media streams.

In this paper, we first evaluate multiple popular and widely used NER methods on the micropost data. The results show a significant decrease of result quality compared to those reported for newswire texts. An interesting observation from the experiment is that we can achieve recall over 90% on the micropost data, when all the results are unified. This means, that in theory, we could achieve very high quality annotations of named entities (NEs) in microposts just by combining existing NER tools in a "smart" way. The rest of the paper is dedicated to the research question, how to combine annotations of different NER tools in order to achieve better recognition in microposts.

We propose an approach for combining NER methods represented by different NE recognizers in order to make a new NE recognizer intended to be used on microposts. The method is designed to combine annotations produced by different NER tools by exploiting machine learning (ML) techniques. We use the term annotation to refer to a substring of an input text that has been marked by a NER tool as a reference to an entity of one of target classes; i.e., LOC, MISC, ORG and PER. The main challenge is the transformation of text annotations produced by NER tools into a

form usable for training ML classification algorithms. Once the NER annotations were transformed to an appropriate format, we have performed an evaluation of a number of popular ML classification techniques. The best performing on our problem domain was the C4.5 algorithm [15] that was used to train decision tree (DT) and random forest (RF) models. The resulting classification model outperformed the best of underlying individual recognizers by more than 10% in F_1 score and a chosen baseline model by 3% in F_1 score.

The main contributions of the work are following: (i) We show that although existing NER tools designed for news text do not perform well on microposts, by merging results of several different NER tools, we can achieve high recall and precision. (ii) We utilize ML classifiers to combine the outputs of multiple NE recognizers. The principal challenge is the transformation of text annotations yielded by NER tools to feature vectors that can be used for the training of classification algorithms. (iii) We provide an extensive evaluation of popular classification models to asses their suitability for the problem of combing results of NER tools. For the best performing ones, we have studied the influence of algorithms parameters on the classification results.

The paper is structured as follows. In Section 2, we briefly summarize research works related to NER. In Section 3 we conduct an experiment, in which a number of existing popular NER tools are evaluated on microposts data. Results show dramatic drop in quality measures compared to the numbers reported on news datasets. In Section 4, we define a baseline NE recognizer, explain our approach of combining NER tools and evaluate our NE recognition models. Finally, Section 5 discusses open issues and Section 6 summarizes our results and concludes the paper.

2. RELATED WORK

There has been a large amount of NER research conducted on formal text, such as newswire or biomedical text. The performance of NE recognizers for this kind of text is comparable to that of humans. For instance, the MUC-7 NE task, where the best NE recognizer scored $F_1 = 93.39\%$, while the annotators scored $F_1 = 97.60\%$ and $F_1 = 96.95\%$ [13]. Another example is CoNLL-2003 shared task, where the best NER recognizer scored $F_1 = 88.76\%$ in English test [22]. It has been later outperformed by Ratinov and Roth [17] achieving $F_1 = 90.8\%$. NE recognizers, which have been designed for these tasks and which achieve state-of-the-art performance results, heavily rely on linguistic features observable in formal text. But many of the important features absent in microposts; e.g. capitalization. Therefore, news-trained recognizers perform worse on them. The performance drop-off is also caused by nature of microposts content – its length, informality, noise and multilingualism. Many of the problems related to NER in microposts are discussed by Bontcheva and Rout in [2].

The idea of combining different methods for NER is not new. It has been successfully applied on formal text by Florian et al. [8], who combine four diverse classifying methods; i.e., transformation-based learning, hidden Markov model, robust risk minimization (RRM) and maximum entropy. Classifiers are complemented by gazetteers together with the output of two externally trained NE recognizers and the whole is used to extract text features. The RMM method is used in order to select a good performing combination of the features. Todorovski and Džeroski [23] introduce meta decision trees (MDT) for combining multiple classifiers. They present a C4.5 algorithm-based training algorithm for producing MDTs. Another application is by Si et al. [21], who combine several NER methods for bio-entity recognition in biomedical texts. They experiment with combining NE classifiers by three different approaches; i.e., majority vote, unstructured exponential model and conditional random field. Also Saha and Ekbal [20] use seven diverse NER classifiers to build a number of voting models depending upon identified text features that are selected mostly without a domain knowledge.

Regarding the NER for tweets, there is also a similar approach taken by Liu et al. [12]. Authors combine a k-Nearest Neighbors (k-NN) classifier with a linear Conditional Random Fields (CRF) model under a semi-supervised learning framework and show increase in F_1 with respect to a baseline system, which is its modified version without k-NN and semi-supervised learning. Etter et al. [6] deal with multilingual NER for short informal text. They do not rely on language dependent features such as dictionaries or POS tagging, but they use language independent features derived from the character composition of a word and its context in a message; i.e., words, character n-grams for words, $\pm k$ words to the left, message length, word length and word position in message. They use an algorithm that combines Support Vector Machine (SVM) with a Hidden Markov Model (HMM) to train a NER model on a manually annotated data. The experiments show that the language independent features lead to F_1 score increase and the model outperforms Ritter et al. [19]. Ritter et al. [19] present re-built NLP pipeline for tweets; i.e., POS tagger, chunker and NE recognizer. The NE recognizer leverages the redundancy inherent in tweets using Labeled LDA [16] to exploit Freebase[1] dictionaries as a source of distant supervision. TwiNER, a novel unsupervised NER system for targeted tweet streams is proposed by Li et al. [11]. Similarly to Etter et al. [6], TwiNER does not rely on any linguistic features of the text. It aggregates information garnered from the Web and Wikipedia. The advantage of TwiNER is that it does not require manually annotated training set. On the other hand, TwiNER does not categorize the type of discovered NEs. Authors prefer the problem of correctly locating and recognizing presence of NEs instead of their classification. Habib and Keulen [9], the winning solution of the #MSM2013 IE Challenge, splits the NER problem in named entity extraction (NEE) and named entity classification (NEC), too. The NEE task is performed by union of entities recognized by two models; i.e., CRF and SVM. Both models are trained on manually labeled tweet data. The CRF involves POS tags and capitalization of the words as features. The SVM segments tweet using Li et al. [11] approach and enriches the segments by external knowledge base (KB). It uses the same features as the SVM model and information from external KB.

3. COMBINED NER METHODS

We have used state-of-the-art NER methods represented by various existing NE recognizers. These methods were combined in our classification models discussed later in this paper. Below we briefly describe used NE recognizers focusing on their NER methods.

1) ANNIE (v7.1) [4] relies on finite state algorithms,

[1] `http://www.freebase.com`

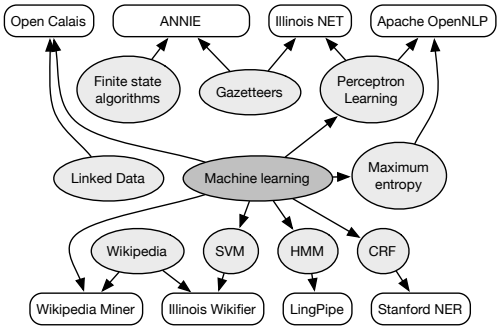


Figure 1: Outline of NE recognizers

gazetteers and the JAPE (Java Annotation Patterns Engine) language. *2) Apache OpenNLP*[2] (v1.5.2) is based on maximum entropy models and perceptron learning algorithm. *3) Illinois Named Entity Tagger* (v1.0.4) [17] uses a regularized averaged perceptron with external knowledge (unlabeled text, gazetteers built from Wikipedia and word class models). We have used Illinois NET with 4-label type set and default configuration. *4) Illinois Wikifier* (v1.0[3]) [18] is based on a Ranking SVM and exploits Wikipedia link structure in disambiguation. *5) Open Calais* operates behind a shroud of mystery since there is not much information available about how its NE recognition works. Official sources[4] say, that it uses NLP, ML and other methods as well as Linked Data. *6) Stanford Named Entity Recognizer* (v1.2.7) [7] is based on CRF sequence models. We have used the English 4-class caseless CONLL model[5]. *7) Wikipedia Miner*[6] [14] is a text annotation tool, which is capable of annotating Wikipedia topics in a given text. It exploits Wikipedia link graph, Wikipedia category hierarchy and relies on ML classifiers, which are used for measuring relatedness of concepts and terms, as well as for measuring disambiguation. We have applied this software to discover Wikipedia topics, which were then tagged according to the DBPedia Ontology[7].

Most of the NE recognizers are based on statistical learning methods. Some of them use also gazetteers and other external knowledge like Wikipedia or Linked Data. Outline of the NE recognizers is depicted in Figure 1.

3.1 NE Recognizers Evaluation

In this section, we evaluate NER methods described in Section 3 on a micropost data corpus. Our intent was to see the performance of each individual NE recognizer. The evaluation was focused also on analysis, which NE recognizer is more suitable for particular named entity class and whether NE recognizers produce diverse results. NE recognizers were evaluated over the adapted #MSM2013 IE Challenge training dataset [1]. We have taken the 1.5 version and cleaned it from duplicate as well as from overlapping microposts with the test dataset. The cleaned training dataset

[2] `http://opennlp.apache.org`
[3] `http://cogcomp.cs.illinois.edu/page/download_view/Wikifier`
[4] `http://www.opencalais.com/about`
[5] english.conll.4class.caseless.distsim.crf.ser.gz
[6] `http://wikipedia-miner.cms.waikato.ac.nz`
[7] `http://dbpedia.org/Ontology`

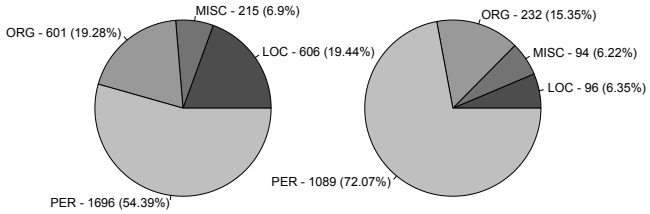


Figure 2: Named entity occurrences in train (left) and test (right) datasets

finally contained 2752 unique manually annotated microposts with classification restricted to four entity types: PER, LOC, ORG and MISC. We have also adapted a test dataset from the #MSM2013 IE Challenge on which we later evaluated our classification models. The occurrence of NEs in both datasets is displayed in Figure 2. Named entity types were not equally distributed. The most frequent entity type in both datasets was PER and the least frequent was MISC. Datasets used in this paper are also available for download[8] in GATE SerialDataStore format. Datasets includes results of all the used NE recognizers as well as our NER models discussed later in the paper.

Evaluated NE recognizers were not specially configured, tweaked or trained for microposts prior to the evaluation. We wanted to see, how they cope with the different kind of text that they were trained for. The alignment with our taxonomy was done by simple mapping. Evaluation results are displayed in Table 1 and ordered by Micro avg. F_1 score. We provide also a Macro summary which averages P, R and F_1 measures on a per document basis, while the Micro summary considers the whole dataset as a one document. The evaluation has also shown, that the NE recognizers produced diverse annotations. This behavior could be seen in raised recall after the results were unified and cleaned from duplicates. Figure 3 illustrates the situation and the possible recall, which could be theoretically achieved when combining the recognizers.

More details about the evaluation can be found in [5]. Some of the evaluation results may slightly differ from those displayed in Table 1. It is because we did accept adjectivals and demonymic forms for countries as *MISC* type in this work; e.g., Americans, English.

4. COMBINING NE RECOGNIZERS

The idea of how to combine NE recognizers was to use ML techniques to build a classification model, which would

[8] `http://ikt.ui.sav.sk/microposts/`

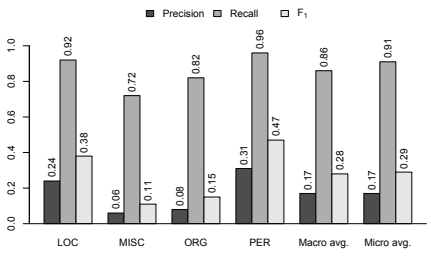


Figure 3: Precision, Recall and F_1 of unified NE recognizers

Table 1: Evaluation of NE recognizers over the training dataset

	F_1				Macro avg.			Micro avg.		
NE recognizer	LOC	MISC	ORG	PER	P	R	F_1	P	R	F_1
OpenCalais	**0.74**	0.26	**0.56**	0.69	0.66	0.50	**0.56**	**0.72**	0.60	**0.65**
Illinois NET	0.72	0.14	0.36	**0.79**	0.50	0.51	0.50	0.61	**0.65**	0.63
Stanford NER	0.67	0.11	0.29	0.75	0.46	0.45	0.46	0.60	0.59	0.60
ANNIE	0.68	–	0.36	0.61	0.71	0.37	0.41	0.64	0.48	0.55
Illinois Wikifier	0.55	0.16	0.51	0.62	0.54	0.42	0.46	0.62	0.47	0.54
Apache OpenNLP	0.51	–	0.27	0.58	0.68	0.28	0.34	0.62	0.38	0.47
Wikipedia Miner	0.56	0.06	0.33	0.61	0.34	**0.52**	0.39	0.32	0.57	0.41
LingPipe	0.35	–	0.07	0.35	0.40	0.30	0.19	0.16	0.38	0.23
Miscinator	–	**0.46**	–	–	**0.92**	0.09	0.12	0.69	0.03	0.05

be trained on features describing microposts' text as well as annotations produced by involved NE recognizers. We have used the training dataset for building the model and the test dataset for evaluating it and comparing with other NE recognizers (Section 3.1).

According to the evaluation results in Section 3.1, we have chosen seven out of eight NE recognizers based on different methods. The discarded one was LingPipe because of its weak[9] performance on micropost data. Chosen NE recognizers were then complemented by *Miscinator*, an NE recognizer specially designed for the #MSM2013 IE Challenge [24].

As overall recall of the underlying NE recognizers was relatively high, we wanted to gain maximum precision while not devalue the recall. We decided to involve ML techniques, but it was necessary to transform this problem into a standard ML task. In this case it was suitable to transform the task of NER into a task of classification. The intent was that ML process would produce a classification model capable of classifying given annotations from involved methods into four target classes LOC, MISC, ORG, PER and one special class NULL indicating that the annotation did not belong to any of the four target classes. Then a simple algorithm would be applied to merge the re-classified annotations into final results.

4.1 Baseline NE Recognizer

We have defined a baseline NE recognizer in the way that each target entity class was extracted by the best NE recognizer according to the evaluation made over the training dataset (section 3.1); i.e., LOC, MISC and ORG classes were extracted by OpenCalais and PER class was extracted by Illinois NET. The performance of the baseline can be seen in Table 2 together with performances of the NE recognizers considered for combining. The evaluation has been made over the test dataset. We can see that the baseline NE recognizer had outperformed underlying NE recognizers in precision and F_1 measure, which was expected. Our goal was to overcome the performance of the baseline NE recognizer with a model produced by ML approach.

4.2 Transforming NEs into Feature Vectors

We have taken an approach of describing how particular methods performed on different entity types compared to the response of other methods and a manual annotation. Used as a training vector, this description was an input for training a classification model. A vector of input training features was generated for each annotation found by underlying NER methods restricted to following types: LOC, MISC, ORG, PER, NP – noun phrase, VP – verb phrase, OTHER – different type. We called this annotation a reference annotation. The vector of each reference annotation consisted of several sub-vectors (Figure 4).

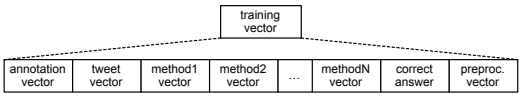


Figure 4: Training vector

The first sub-vector of the training vector was an annotation vector (Figure 5). The annotation vector described the reference annotation – whether it was upper or lower case, used a capital first letter or capitalized all of its words, the word count, and the type of the detected annotation.

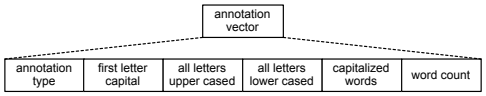


Figure 5: Annotation vector

The second sub-vector described microposts as a whole (Figure 6). It contained features describing whether all words longer than four characters were capitalized, uppercase, or lowercase. We called this sub-vector tweet vector.

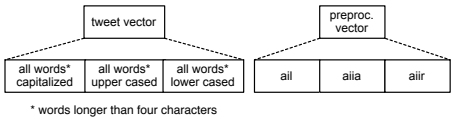


Figure 6: Tweet vector (left) and preprocessing vector (right)

The rest of the sub-vectors were computed according to the overlap of the reference annotation with annotations produced by particular NER method. Such sub-vector (termed a method vector by us) was computed for each method and contained four other vectors describing the overlap of method annotations with reference annotation on each target entity type (Figure 7). The *annotation type* attribute was filled with a class of method annotation that exactly matched position of the reference annotation and was one of the target entity classes, otherwise it was left blank.

Each overlap vector of a particular method and NE class (Figure 8) consisted of five components – *ail*: the average intersection length of a reference annotation with the method

Table 2: Evaluation of NE recognizers over the test dataset

	F_1				Macro avg.			Micro avg.		
Model	LOC	MISC	ORG	PER	P	R	F_1	P	R	F_1
Baseline	**0.61**	**0.29**	0.30	**0.84**	0.69	0.44	**0.51**	**0.83**	0.67	**0.74**
Illinois NET	0.50	0.06	0.32	**0.84**	0.41	**0.46**	0.43	0.65	**0.69**	0.67
Stanford NER	0.51	0.00	0.30	0.82	0.39	0.43	0.41	0.67	0.67	0.67
Open Calais	**0.61**	**0.29**	0.30	0.69	0.64	0.41	0.47	0.66	0.60	0.63
ANNIE	0.48	–	0.19	0.68	0.61	0.32	0.34	0.63	0.52	0.57
Illinois Wikifier	0.34	0.09	**0.46**	0.68	0.44	0.38	0.39	0.63	0.50	0.55
Apache OpenNLP	0.38	–	0.13	0.64	0.57	0.27	0.29	0.62	0.43	0.51
Wikipedia Miner	0.29	0.04	0.29	0.67	0.28	**0.46**	0.32	0.32	0.57	0.41
LingPipe	0.15	–	0.05	0.38	0.37	0.28	0.14	0.15	0.38	0.21
Miscinator	–	0.19	–	–	**0.88**	0.03	0.05	0.52	0.01	0.01

[9] we have used the *English News: MUC-6* model

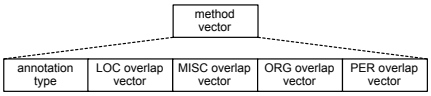


Figure 7: Method vector

annotations of the same NE class, *aiia*: the average intersection ratio of the method annotations of the same NE class with reference annotation, *aiir*: the average intersection ratio of a reference annotation with method annotations of the same NE class, *average confidence* (if the underlying method return such value), and *variance of the average confidence*.

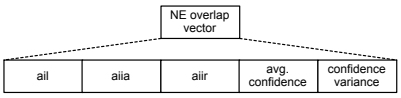


Figure 8: Overlap vector

The *ail* component in overlap vector was computed using formula (1), where R was a fixed reference annotation and M_C was a set of n method annotations of class C intersecting with the reference annotation R. The *ail* component was a simple arithmetic mean of intersection lengths.

$$ail_{(R,M_C)} = \frac{1}{n}\sum_{i=1}^{n}|R \cap M_{Ci}| \quad (1)$$

The *aiia* component was computed using formula (2), which was also a simple arithmetic mean, but the intersection lengths were normalized by lengths of particular method annotations M_{Ci} intersecting with the reference annotation R. We wanted the value of *aiia* component to describe how much were method annotations covered by the reference annotation.

$$aiia_{(R,M_C)} = \frac{1}{n}\sum_{i=1}^{n}\frac{|R \cap M_{Ci}|}{|M_{Ci}|} \quad (2)$$

Similarly, the *aiir* component was computed using formula (3), but the intersection lengths were normalized by length of the reference annotation R. The value of *aiir* component was used to describe how much was the reference annotation covered by method annotations.

$$aiir_{(R,M_C)} = \frac{1}{n}\sum_{i=1}^{n}\frac{|R \cap M_{Ci}|}{|R|} \quad (3)$$

A simple example of overlap vector computation is depicted in Figure 9. The overlap vector is computed for method 4 and PER class according to the highlighted reference annotation. In this example, the reference annotation is M2.PER1, but it can be any method annotation or manual annotation. The rest of the method 4 overlap vectors are zero-valued since method 4 does not return annotations of types LOC, MISC and ORG. Similarly, there will be overlap vectors according to the same reference annotation computed for methods 1, 2 and 3 to finally have all method vectors computed in a training vector. In addition, there will be eight training vectors computed, because of eight annotations taken as reference annotations, where also the manual annotation PER is included.

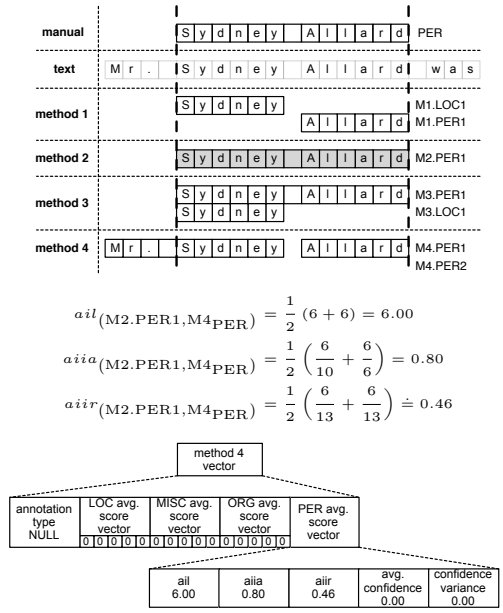


Figure 9: Example of overlap vector computation

The last two components in the training vector were the correct answer (i.e., the correct annotation type taken from manual annotation) and a special preprocessing vector (Figure 6). The preprocessing vector included three components: *ail*, *aiia* and *aiir*, which described the intersection of the reference annotation when it was correct with the correct answer. If the reference annotation was not correct the values of the preprocessing vector components were set to zero.

The number of learning features depended on the number of combined methods, since for each involved method a new method vector was computed and included into the training vector. There were some features, which were less or more important or not important at all. The effect of specific learning features is discussed later.

4.3 Training Data Preprocessing

Training data was generated automatically as a collection of training vectors, which needed further processing prior to apply ML algorithms. There have been duplicate training vectors removed in order to eliminate distortion in training and validation process thus getting a more balanced classification model.

According to the preprocessing vector (Figure 6), there have been training vectors removed, in which the *annotation type* attribute in the *annotation vector* was correct but the *aiir* attribute in the preprocessing vector was not equal to 1.0, i.e., the bounds of the reference annotation were not equal to the bounds of the correct answer. In previous versions, we tried to accept all the training vectors whose *aiir* attribute was at least 0.95, i.e., the reference annotation overlapped with the correct answer at least on 95%, but this led to models with lower precision.

We have removed also several attributes, which led to zero information gain and which were not useful for the classification, i.e., attributes with the same value for all the training vectors. They were usually *average confidence* and *variance of the average confidence* scores, because some NE recog-

Table 3: Performance of classification models built by different algorithms

Model	AUROC	ACC	F_1
Decision Tree J48	**0.939**	0.969	**0.938**
Random Forest	0.927	**0.972**	0.925
Bagging	0.912	**0.972**	0.908
Multilayer Perceptron	0.895	0.955	0.890
Dagging	0.889	0.922	0.880
Bayess Net	0.857	0.954	0.865
RBF Network	0.850	0.923	0.835
AdaBoost.M1	0.811	0.804	0.750
Naive Bayes	0.797	0.919	0.814

nizers did not provide annotation confidence information, hence both attributes were always zero and therefore also their information gain. Due to same reasons, we have removed also attributes, which contained information in less than 3% of records. Attributes of the *preprocessing vector* have been also removed.

The preprocessing phase had significantly reduced the size of training data and therefore memory requirements as well as it had sped up the training process. It started with a set of $\sim 63,000$ training vectors with ~ 200 attributes and finished on $\sim 31,000$ unique records with ~ 100 highly relevant attributes.

4.4 Model Training and Evaluation

We have tried several algorithms to train different classification model candidates, which we compared according to the F_1 score. We have also examined AUROC and ACC (accuracy) measures. All these three measures were obtained from 10-fold cross validation of the model candidates over the training dataset. Cross validation served as a good method for identifying suitable model candidates, because it avoided an effect of overfitting without a need of another test dataset. The best performance has been achieved by DT classification model built with J48[10] algorithm (DTJ48) followed by RF [3] model. The third was a classification model based on REPTree (Reduced Error Pruned Tree) built with Bagging algorithm (Table 3). We have focused on the first two best performing algorithms and built several classification models while varying some of input parameters of these algorithms in order to gain precision and recall. It was *Minimum Number of Instances per Leaf* parameter (hereinafter parameter "M") for DTJ48 and *number of trees* for RF. The classification models were evaluated using a hold-out validation method over the test dataset. Evaluation results are displayed in Table 4. The best performing were models based on RF, which outperformed models based on DT, baseline recognizer and all the underlying NE recognizers. We can see that recall and precision have been growing with the number of trees in the RF models and continued to converge to 79% and 76% respectively. This behavior is more obvious in Figure 10, where F_1 measures are depicted for particular NE classes according to the variated number of trees. Dashed lines indicate score of the baseline model.

Evaluation results of models built with J48 algorithm (C4.5), while varying the M parameter, are displayed in Figure 11. We can see that the F_1 score for LOC has been approaching the baseline score similarly as it was for RF algorithm while varying the number of trees parameter. Analogous behavior can be seen in Macro and Micro average scores. In ORG and PER classification the score was higher

[10]J48 is an implementation of C4.5 algorithm

Table 4: Evaluation of classification models over the test dataset

	F_1				Macro avg.			Micro avg.		
Model	LOC	MISC	ORG	PER	P	R	F_1	P	R	F_1
RF N400	0.60	0.23	**0.49**	**0.88**	0.60	0.53	**0.55**	0.79	**0.76**	**0.77**
RF N300	0.60	0.23	**0.49**	**0.88**	0.60	0.53	**0.55**	0.79	**0.76**	**0.77**
RF N200	0.59	0.24	0.48	**0.88**	0.60	0.53	**0.55**	0.79	**0.76**	**0.77**
RF N100	0.58	0.23	0.48	**0.88**	0.59	0.53	0.54	0.79	**0.76**	**0.77**
RF N9	0.57	0.26	0.47	0.87	0.55	**0.54**	0.54	0.76	**0.76**	0.76
RF N21	0.55	0.26	0.47	0.87	0.56	0.53	0.54	0.77	**0.76**	0.76
RF N17	0.56	0.26	0.48	**0.88**	0.56	**0.54**	0.54	0.77	**0.76**	0.76
RF N14	0.55	0.26	0.46	**0.88**	0.55	0.53	0.54	0.76	**0.76**	0.76
RF N11	0.57	0.25	0.46	0.87	0.56	0.53	0.54	0.76	**0.76**	0.76
DTJ48 M13	0.57	**0.36**	0.36	0.87	0.60	0.52	0.54	0.78	0.73	0.75
RF N7	0.56	0.25	0.44	0.87	0.53	0.53	0.53	0.75	**0.76**	0.75
DTJ48 M11	0.59	0.27	0.40	0.86	0.56	0.51	0.53	0.77	0.73	0.75
DTJ48 M9	0.55	0.29	0.39	0.86	0.55	0.51	0.52	0.77	0.72	0.75
DTJ48 M7	0.57	0.23	0.41	0.86	0.53	0.51	0.52	0.75	0.73	0.74
RF N5	0.53	0.22	0.42	0.86	0.51	0.52	0.51	0.73	0.75	0.74
Baseline	**0.61**	0.29	0.30	0.84	**0.69**	0.44	0.51	**0.83**	0.67	0.74
DTJ48 M5	0.54	0.23	0.43	0.85	0.53	0.51	0.51	0.75	0.72	0.74
#MSM2013 21_3	0.50	0.31	0.41	0.83	0.51	0.53	0.51	0.70	0.73	0.71
DTJ48 M2	0.45	0.31	0.37	0.84	0.50	0.49	0.49	0.71	0.71	0.71
RF N3	0.50	0.20	0.37	0.85	0.46	0.50	0.48	0.68	0.73	0.71
RF N2	0.51	0.15	0.33	0.84	0.44	0.49	0.46	0.64	0.71	0.68

than the baseline or at least the same. We cannot say, that it has been growing with the parameter M. The same applies for MISC, where the F_1 score varied around the baseline. In general, increasing minimum number of instances per leaf in DT (parameter M) led to models with higher recall and precision. There were four classification models, which have slightly outperformed the baseline model, but not as much as the RF models.

The #MSM2013 21_3 model in the Table 4 is our submission to the #MSM2013 IE Challenge [24]. This model was one of our early models, which were based on groundwork of this paper. The model has finished on the second place in the challenge loosing 1% in F_1 on a winner Habib et. al [9]. Results of this model in the table may be slightly worse than the official challenge results[11], since we have used more strict evaluation criteria. We did not accept partially correct consecutive annotations; i.e., PER/Christian PER/Bale was incorrect, while PER/Christian Bale was correct. For a better

[11]`http://oak.dcs.shef.ac.uk/msm2013/ie_challenge/results/challenge_results_summary.pdf`

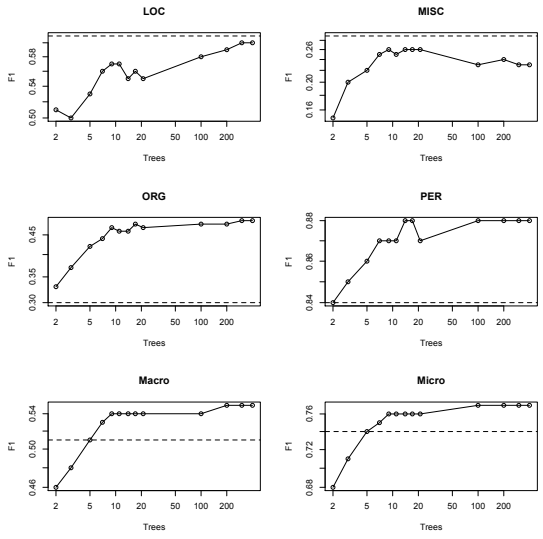


Figure 10: Impact on F_1 while varying number of trees for Random Forest algorithm

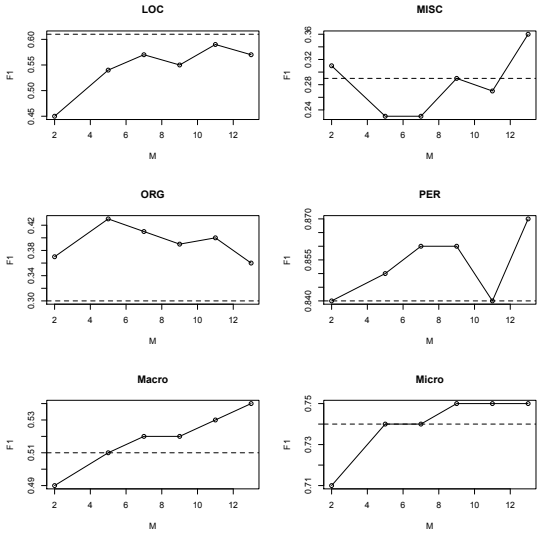


Figure 11: Impact on F_1 while varying parameter M for Decision Tree J48 (C4.5) algorithm

comparison of the models, we present precision, recall and F_1 measures of the best performing model – RF N400, best DT model – DTJ48 M13, baseline recognizer and the three best performing NE recognizers in Figure 12. The gain in precision of the RF N400 model with respect to the NE recognizer with the highest precision – Stanford NER was 18%. However, the baseline recognizer had higher precision than RF N400 by 4%. Model based on DT – DTJ48 M13 was the third best in precision followed by Stanford NER. The highest score in recall among the combined NE recognizers has been achieved by Illinois NET reaching 69%. The gain in recall of the RF N400 model with respect to Illinois NET was 10%. RF N400 reached the highest score in recall followed by DTJ48 M13 and Illinois NET. Stanford NER and the baseline recognizer shared the fourth place.

The highest score in F_1 measure among the combined NE recognizers has been achieved by Illinois NET and Stanford NER, which both reached 67%. The gain in F_1 of RF N400 with respect to them was 15%. RF N400 model with 400 trees has outperformed also the second DTJ48 M13 model and the third baseline recognizer, whose gain was 10%. A comparison on NE class basis is depicted in Figure 13. We did not include the baseline recognizer in the charts, since it is represented there by its NE recognizers (see Section 4.1). Our RF N400 model was the best in recognizing two most occurring entity classes in the test dataset – ORG and PER. It has gained 7% and 5% with respect to Illinois Wikifier and Illinois NET respectively. The best in recognizing LOC entities was Open Calais, on which the RF N400 model lost 1%. The MISC entity type was a domain of the DTJ48 M13 model, which has gained 24% with respect to the second Open Calais.

Closer analysis of annotation results has shown, that there have been many results correctly classified, but they did not exactly match position in text; i.e., results were partially correct. Therefore we tried to apply post-processing and trimmed non-alphabetical characters off the results. We have also removed definite articles from LOC and PER results. Moreover, we have removed titles from PER results; e.g., Dr., Mr. or Sir. Evaluation of models with this simple post-processing (PP) is displayed in Table 5. We have applied post-processing on the best versions of RF and DT models. The gain in F_1 with respect to models without post-processing was 3%. Finally, we tried to build up a model by combining our best models, which were RF N400 PP for LOC, ORG, PER NE classes and DTJ48 M13 PP for MISC class. This model had better performance in MISC recognition, but the overall improvement was not markable, because the occurrence of MISC entities in the test dataset was very low, thus it did not significantly affect the F_1 score.

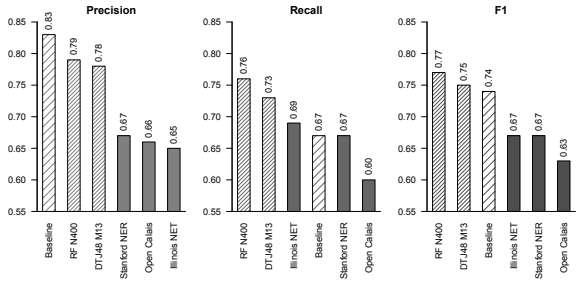


Figure 12: Comparison of the three best NE recognizers with the baseline recognizer and our two best performing models RF N400 and DTJ48 M13

Table 5: Evaluation of classification models using post-processing (PP) over the test dataset

	F_1				Macro avg.			Micro avg.		
Model	LOC	MISC	ORG	PER	P	R	F_1	P	R	F_1
C4.5M13 PP + RF N400 PP	**0.61**	**0.36**	**0.56**	**0.88**	0.66	**0.58**	**0.60**	0.80	**0.78**	**0.79**
RF N400 PP	**0.61**	0.25	**0.56**	**0.88**	0.63	0.55	0.58	0.80	0.77	**0.79**
DTJ48 M13 PP	0.58	**0.36**	0.44	**0.88**	0.63	0.54	0.56	0.80	0.75	0.77
RF N400	0.60	0.23	0.49	**0.88**	0.60	0.53	0.55	0.79	0.76	0.77
DTJ48 M13	0.57	**0.36**	0.36	0.87	0.60	0.52	0.54	0.78	0.73	0.75
Baseline	**0.61**	0.29	0.30	0.84	**0.69**	0.44	0.51	**0.83**	0.67	0.74
#MSM2013 21_3	0.50	0.31	0.41	0.83	0.51	0.53	0.51	0.70	0.73	0.71

5. DISCUSSION AND FUTURE WORK

The structure of the best models (DTJ48 M13 and RF N400) is based on DTs, which use rules always related to one input attribute. This could present a weakness of

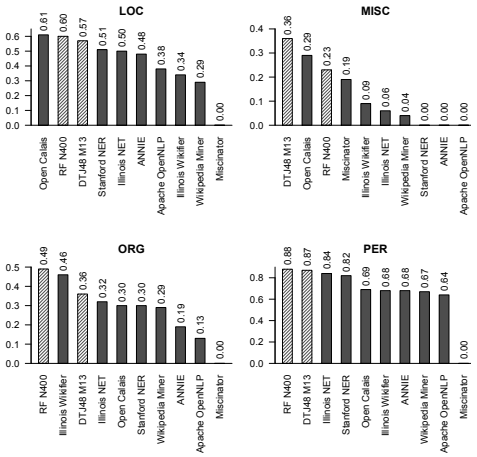


Figure 13: Comparison of the combined NE recognizers with our two best performing models RF N400 and DTJ48 M13 by F_1 and NE class

these models. One possible solution could be to use multivariate DTs, which support multiple attributes per node in a tree and can handle also correlated attributes [10]. The drawback of using multivariate DTs is in the time needed to built them, but on the other hand their time performance is higher, because they do not test the same attribute multiple times. We expect that such models could better utilize the potential of data and therefore could be also more accurate than RF or DT models.

6. CONCLUSIONS

We have shown an approach of combining NE recognizers based on diverse methods on a task of NER in microposts and examined several ML techniques for the combination of text and annotation features produced by the recognizers. The best performing were RF and DT based on C4.5 algorithm. Combination models produced by these algorithms have achieved performance superior to that of underlying NE recognizers as well as the baseline recognizer, which was built of the best performing NE recognizers for each target NE class. The best of our combination models was RF N400, an RF model with 400 trees. Its gain in F_1 with respect to the best individual NE recognizer was 15% and with respect to the baseline recognizer 4%. Performance of the RF and DT models indicated that ML techniques lead to more favorable combination of underlying NE recognizers than it was done manually in the baseline NE recognizer. The advantage of the ML models is that they can adapt to actual text according to its features and annotations from underlying NE recognizers, as well as benefit from given negative examples.

7. ACKNOWLEDGMENTS

This work was supported by projects VEGA 2/0185/13, VENIS FP7-284984 and CLAN APVV-0809-11.

8. REFERENCES

[1] A. E. C. Basave, A. Varga, M. Rowe, M. Stankovic, and A.-S. Dadzie. Making sense of microposts (#msm2013) concept extraction challenge. In *Making Sense of Microposts (#MSM2013) Concept Extraction Challenge*, pages 1–15, 2013.

[2] K. Bontcheva and D. Rout. Making sense of social media streams through semantics: a survey. *Semantic Web*, 2012.

[3] L. Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, Oct. 2001.

[4] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. ACL'02. ACL, 2002.

[5] S. Dlugolinsky, M. Ciglan, and M. Laclavik. Evaluation of named entity recognition tools on microposts. INES 2013. IEEE, 2013.

[6] D. Etter, F. Ferraro, R. Cotterell, O. Buzek, and B. Van Durme. Nerit: Named entity recognition for informal text. Technical report, Technical Report 11, HLTCE, Johns Hopkins University, July 2013.

[7] J. R. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. ACL '05, pages 363–370, Stroudsburg, PA, USA, 2005. ACL.

[8] R. Florian, A. Ittycheriah, H. Jing, and T. Zhang. Named entity recognition through classifier combination. CONLL '03, pages 168–171, Stroudsburg, PA, USA, 2003. ACL.

[9] M. Habib, M. V. Keulen, and Z. Zhu. Concept extraction challenge: University of Twente at #msm2013. In *Making Sense of Microposts (#MSM2013) Concept Extraction Challenge*, pages 17–20, 2013.

[10] T. S. Korting. C4. 5 algorithm and multivariate decision trees, image processing division. *National Institute for Space Research–INPE São José dos Campos–SP, Brazil*, 2006.

[11] C. Li, J. Weng, Q. He, Y. Yao, A. Datta, A. Sun, and B.-S. Lee. Twiner: Named entity recognition in targeted twitter stream. SIGIR '12, pages 721–730, New York, NY, USA, 2012. ACM.

[12] X. Liu, S. Zhang, F. Wei, and M. Zhou. Recognizing named entities in tweets. HLT '11, pages 359–367, Stroudsburg, PA, USA, 2011. ACL.

[13] E. Marsh and D. Perzanowski. Muc-7 evaluation of ie technology: Overview of results. MUC-7, April 1998.

[14] D. Milne and I. H. Witten. An open-source toolkit for mining wikipedia. *Artif. Intell.*, 194:222–239, 2013.

[15] J. R. Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[16] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. EMNLP '09, pages 248–256, Stroudsburg, PA, USA, 2009. ACL.

[17] L. Ratinov and D. Roth. Design challenges and misconceptions in named entity recognition. CoNLL '09, pages 147–155. ACL, 2009.

[18] L. Ratinov, D. Roth, D. Downey, and M. Anderson. Local and global algorithms for disambiguation to wikipedia. HLT '11, pages 1375–1384. ACL, 2011.

[19] A. Ritter, S. Clark, Mausam, and O. Etzioni. Named entity recognition in tweets: An experimental study. EMNLP '11, pages 1524–1534, Stroudsburg, PA, USA, 2011. ACL.

[20] S. Saha and A. Ekbal. Combining multiple classifiers using vote based classifier ensemble technique for named entity recognition. *Data Knowl. Eng.*, 85:15–39, May 2013.

[21] L. Si, T. Kanungo, and X. Huang. Boosting performance of bio-entity recognition by combining results from multiple systems. BIOKDD '05, pages 76–83, New York, NY, USA, 2005. ACM.

[22] E. F. Tjong Kim Sang and F. De Meulder. Introduction to the conll-2003 shared task: language-independent named entity recognition. CONLL '03, pages 142–147, Stroudsburg, PA, USA, 2003. ACL.

[23] L. Todorovski and S. Džeroski. Combining classifiers with meta decision trees. *Machine Learning*, 50(3):223–249, 2003.

[24] Štefan Dlugolinský, P. Krammer, M. Ciglan, and M. Laclavík. MSM2013 IE Challenge: Annotowatch. In *Making Sense of Microposts (#MSM2013) Concept Extraction Challenge*, pages 21–26, 2013.

HG-RANK: A Hypergraph-based Keyphrase Extraction for Short Documents in Dynamic Genre

Abdelghani Bellaachia
Department of Computer Science
The George Washington University
Washington, DC 20052, USA
bell@gwu.edu

Mohammed Al-Dhelaan*
Department of Computer Science
The George Washington University
Washington, DC 20052, USA
mdhelaan@gwu.edu

ABSTRACT

Conventional keyphrase extraction algorithms are applied to a fixed corpus of lengthy documents where keyphrases distinguish documents from each other. However, with the emergence of social networks and microblogs, the nature of such documents has changed. Documents are now of short length and evolve topics which require specific algorithms to capture all features. In this paper, we propose a hypergraph-based ranking algorithm that models all the features in a random walk approach. Our random walk approach uses weights of both hyperedges and vertices to model short documents' temporal and social features, as well as discriminative weights for word features respectively, while measuring the centrality of words in the hypergraph. We empirically test the effectiveness of our approach in two different data sets of short documents and show that our approach has an improvement of 14% to 25% in precision over the closest baseline in a Twitter data set and 10% to 27% in the Opinosis data set.

Categories and Subject Descriptors

I.2.7 [**ARTIFICIAL INTELLIGENCE**]: Natural Language Processing—*Text analysis*

Keywords

Text hypergraphs; Keyphrase extraction; Random walks; Short documents; Hypergraph random walks

1. INTRODUCTION

Short text messages are ubiquitous nowadays in social networks and across the web. Regardless of the length limitation, the size restriction did not limit the popularity of social interaction through such messages. Twitter, for instance, has more than 200 million active users each month[1]. Such high popularity necessitates ranking systems capable of measuring the importance of keywords and keyphrases within such limited length to facilitate search, indexing, and detecting trends. By finding salient terms, tasks such as summarization and text visualization become feasible. However, the dynamic nature of social microblogs makes ranking a non-trivial task.

Microblogs have a dynamically changing content that needs specially designed algorithms for keyphrase extraction. Descriptive keyphrases are keyphrases that signify topics in a document and help differentiate it from other documents in the corpus. However, the social aspects and evolution of topics in a social media genre make it rather difficult to find keyphrases. Most keyphrase extraction algorithms do not account for the temporal and social attributes when finding keyphrases since they are designed for static documents corpora. Therefore, a number of interesting research questions arise in social microblogs where topics change frequently. If the content is dynamically changing, then can we rely on conventional keyphrases extraction approaches? How can we account for the temporal and social attributes in social media for keyphrase extraction?

In this article, we present a hypergraph-based algorithm, called HG-RANK, that is capable of modeling temporal and social aspects in addition to discriminative weights. A hypergraph is a generalization of graphs were edges have a set of vertices (called hyperedges) instead of two nodes. We define a lexical hypergraph where vertices are distinct words and hyperedges are short documents that contain the words. We model the temporal and social attributes of documents as hyperedge weights to reflect the attributes over the document's keywords, and we model discriminative term weights as vertex weights to give the model the ability of recognizing topical terms. We design a weighted random walk over the hypergraph to measure the centrality of keywords taken into account all the aforementioned features.

To rank vertices in a hypergraph, we generalize a probabilistic random walk suitable for a weighted hypergraph structure. The surfer considers the weights of both vertices and hyperedges for transitioning. The intuition is that the surfer will prefer words that has the following properties. The words belong to a recent document, and they exist in a document that has attracted social users, for in-

*This author is sponsored by King Saud University, Saudi Arabia

Published as part of the #Microposts2014 Workshop proceedings, available online as CEUR Vol-1141 (http://ceur-ws.org/Vol-1141)

#Microposts2014, April 7th, 2014, Seoul, Korea.

[1]https://twitter.com/twitter/status/281051652235087872

stance re-tweet in Twitter. Additionally, the surfer will prefer topically discriminative words capable of finding accurate keyphrases.

The contribution of this paper can be summarized as follows:

- We propose a new hypergraph approach to jointly model temporal and social features within the hypergraph structure. This model is capable of recognizing the importance of time and social features that are important in a dynamic genre.
- The hypergraph-based HG-RANK algorithm is the first graph-based approach for keyphrase extraction that considers the high-order relation between words instead of a pair-wise relation as in conventional graph-based keyphrase extraction.
- We evaluate our approach with two different data sets Twitter and Opinosis. We show the effect of each dimension on the task of keyphrase extraction.

The rest of the paper is organized as follows. A discussion of the related work is in Section 2. We define the hypergraph notation needed for explaining the proposed approach in Section 3. The proposed approach will be thoroughly explained in Section 4. Section 5 will describe the data and experimental results. The paper conclusion is in Section 6.

2. RELATED WORK

Our work is related to three different research areas, namely: temporal and social aspects for keyphrase extraction, graph-based keyphrase extraction, and hypergraphs. This work bridges such areas for the task of keyphrase extraction in a unified framework.

The emergence of social networks has motivated researchers to examine the inclusion of temporal and social dimensions into search [27][16][13], summarization[24][23], and keyphrase extraction[28][11]. Yu et al. proposed to combine the temporal dimension into a PageRank[6] approach for ranking research publications considering their publication time[27]. Wan proposed a time-aware summarization algorithm over a lexical graph[24]. A probabilistic approach for personalized temporally-aware tweets summarization is proposed in [22] For including social aspects, Zhao et al. proposed to do keyphrase extraction while they used an interestingness score for capturing social attribute[28]. A multi-document summarizer that takes into account social features is proposed in [17]. Moreover, a lexical graph expansion for extracting keyphrases through social hashtags is shown in [2].

A number of graph-based keyphrase extraction approaches have been proposed. TextRank[20], LexRank[9], NE-Rank[3], SingleRank[25], and TopicRank[5]. These algorithms leverage a random walk to calculate the centrality of either words or sentences. For instance, NE-Rank considers node weights being tf-idf of words and edge weight being freqancy of co-occurrence of pairs of words. However, they all use simple graphs not hypergraphs. In this paper, we consider a high-order co-occurrence relation modeled in a hypergraph.

Hypergraph random walks have been proposed in [29][1]. We further extend the aforementioned approaches in this work to include vertex weights. Wang et al. proposed to use a semi-supervised ranking approach based on Zhou et al.[29] for ranking sentences which they used for text summarization[26]. Li et al.[14][15] proposed a semi-supervised keyphrase ranking over hypergraphs based on Zhou et al.[29] defintions. They proposed using semantic connection between phrases(vertices) to form hyperedges using external knowledge sources as in Wikipedia. Our work is different in the following matter: we use a completely unsupervised approach for ranking keywords instead of sentences or phrases which may not be easy to find in social snippets, we propose a new weighted random walk that uses both hyperedges and vertices weights, and we include temporal and social attributes in the ranking. Finally, unlike exciting approaches for semi-supervised hypergraph ranking our ranking approach is query independent and thus unsupervised.

3. NOTATIONS AND DEFINITIONS

Let $HG(V, \mathcal{E})$ be a hypergraph with the vertex set V and the set of hyperedges $\mathcal{E}$. A hyperedge e is a subset of V where $\cup_{e \in \mathcal{E}} e = V$. Let $HG(V, \mathcal{E}, w)$ be a weighted hypergraph where $w : \mathcal{E} \rightarrow \mathbb{R}^+$ is the hyperedge weight. A hyperedge e is said to be incident with v when $v \in e$. A hypergraph has an incidence matrix $H \in \mathbb{R}^{|V| \times |E|}$ as follows:

$$h(v, e) = \begin{cases} 1 & \text{if } v \in e \\ 0 & \text{if } v \notin e \end{cases} \quad (1)$$

The vertex and hyperedge degree are defined as follows:

$$d(v) = \sum_{e \in E} w(e) h(v, e) \quad (2)$$

$$\delta(e) = \sum_{v \in V} h(v, e) = |e| \quad (3)$$

D_e and D_v are the diagonal matrices representing the degrees of hyperedges and vertices, respectively. W_e is the diagonal matrix with the hyperedge weights.

4. PROPOSED APPROACH

The HG-RANK model captures keyphrases using a hypergraph structure where it is possible to inherently model social and temporal features. These features are embedded as a hyperedge weight that represents a specific short document. In essence, we model each short text document d_i as a bag-of-words model with distinct keywords $d_i = \{k_1, k_2, ..., k_s\}$. A collection of documents $D_i = \{d_1, d_2, ..., d_n\}$ is then represented as a lexical hypergraph in the following manner. We represent each short document as a hyperedge, and each keyword as a distinct vertex.

In this section, we will describe the HG-RANK algorithm in more depth. First, the calculation and insertion of the temporal and social attributes is going to be explained. Second, the vertex weights will be explained. Third, we will define the random walk ranking approach on the lexical hypergraph to rank keywords. Finally, we will discuss our approach on extracting keyphrases.

4.1 Modeling Temporal & Social Features

Temporal attributes in a dynamic genre as a microblogging social network or news trends is an important dimension to understand evolving topics and keyphrases. We measure the temporal effect as a ranking function for short documents. The more recent the document, the higher the temporal rank will be. Similar to [27][24], we measured the

temporal effect as the following:

$$R_{time}(d_i) = Q^{(c-y_i)/24} \quad (4)$$

Where c and y_i denote the current time and the document d_i publication time, respectively. $(c-y_i)$ is the time interval between current and publication time in hours. We divide by 24 to show the difference of publication time and current time in number of days. Q is a decay rate parameter with values $0 < Q < 1$. Moreover, the Q value is inversely proportional to favoring recent documents. When Q is closer to 0, the ranking favors very recent documents over old ones. On the other hand, when Q is closer to 1, the ranking is less focused on new documents. In our experiments, we set Q to 0.5.

For the social effect, we measure the social dimension of documents as a ranking function. The more popular or shared the document, the higher the social rank will be. For example in Twitter, tweets that are re-tweeted frequently should be more important than a tweet without re-tweets. This is similar to other social networks with the "like" feature as in Facebook or product reviews. We calculate the social ranking as follows:

$$R_{social}(d_i) = \frac{s_i + 1}{\sum_e s_e + 1} \quad (5)$$

Where s_i is the counter of social feature (counts of re-tweet or likes) for document d_i. $\sum_e s_e$ is the sum of all social features across all documents (total number of re-tweets for example). Moreover, we added one smoothing to avoid canceling out documents with no social attributes.

Now we tie both temporal and social features together in one ranking function as follows:

$$w(d_i) = \lambda R_{social}(d_i) + (1-\lambda) R_{time}(d_i) \quad (6)$$

λ is a smoothing parameter with $0 < \lambda < 1$ to trade off the effect of temporal aspects and social aspects. We experimented with different values for λ which will be discussed in the experiment section. The final documents rank $w(d_i)$ will be embedded in the hypergraph as a *hyperedge weight* to reflect documents' importance over keywords. The intuition behind embedding temporal and social features in the ranking scheme is that they are essential for capturing keyphrases in a dynamic genre. In a dynamic genre, as in Twitter, the content rapidly changes with time. Hence, the keyphrases tend to change as well. Conventional keyphrase extraction algorithms do not consider the time dimension in finding keyphrases which make them insufficient for the task. Moreover, the social aspect is important to capture keyphrases of trendy topics that social network users find interesting. An interesting topic in social media will more likely be searched compared to other topics which makes it important to find its keyphrases. We will discuss vertex weights in the next section.

4.2 Modeling Discriminative Weights as Vertex Weights

Graph-based approaches base the ranking on the relational structure of co-occurring words. Such ranking is great on capturing the semantic relation between words. However, there is no evidence that graph-based ranking approaches are able to capture discriminative words. To enhance the hypergraph-based ranking algorithm, we use a discriminative weighting scheme tf-idf as vertex weights before we start the random walk. This injection of tf-idf weights will add a discriminative perspective for calculating the rank though a random walk approach. However, when applied to short text documents, tf-idf fails to capture descriptive terms due to sparsity of features (short length). To circumvent the sparsity problem, we aggregate short documents to a virtual larger ones and then calculate tf-idf scores. A larger virtual document δ is the concatenation of smaller documents d which is $\delta_t = \{d_1 + d_2 + ... + d_n\}$. In Section 5.3, different approaches for aggregation are described in more depth. We measure the normalized tf-idf over the larger documents being the set of $D = \{\delta_1, \delta_2, ..., \delta_n\}$. The tf-idf is measured as follows:

$$w(v_i)_{tf\text{-}idf} = \frac{tf(v_i)}{N_w} \cdot log \frac{N}{df(v_i)} \quad (7)$$

Where $tf(v_i)$ as the term frequency on the document δ and N_w is sum of all words occurrences in document δ for normalization. N is the number of documents in the larger document set D, and $df(v_i)$ is the number of larger documents in D that contain the term v_i. We will discuss the hypergraph ranking algorithm HG-Rank in detail in the next section.

4.3 HG-Rank: Ranking in a Hypergraph

To rank vertices in a hypergraph, we generalize a random walk process for hypergraphs. A random walk process is the transitioning between vertices in a graph by starting at a given vertex and moving to another neighboring vertex after each discrete time step t. We can imagine vertices as a set of states $\{s_1, s_2, ..., s_n\}$ and the transitioning to be a finite Markov chain $\mathcal{M}$ over these states. The transition probability calculated as $P(u,v) = Prob(s_{t+1} = v | s_t = u)$ which means that the chain $\mathcal{M}$ will be at v at time $t+1$ given that it was observed at u at time t. The Markov chain herein is *homogeneous* which means that the transition probability is independent of time t. Note that for any vertex u we have $\sum_v P(u,v) = 1$. Since $\mathcal{M}$ is homogeneous with probabilities computed over only a single transition, we can then define a transition matrix $P \in \mathbb{R}^{|V| \times |V|}$ for all moves. The transition matrix P captures the transition between vertices which shows the behavior of a surfer randomly moving between vertices according to such probabilities. Next we will show how we define the random walk in hypergraphs.

In simple graphs[2], the random walk process is clear by simply choosing an edge with a probability to a destination vertex. However, it is not the case in hypergraphs where the structure of the graph is substantially different demanding a more general walk. For instance, in a hypergraph, a hyperedge could have more than two end-point vertices $\delta(e) \geq 2$. To generalize the random walk process in hypergraphs, we model the walk as the transition between two vertices that are incident to each other in a *hyperedge* instead of a normal edge. In essence, the random walk is seen to be a two-step process, instead of one, which is the following: the random surfer first chooses a hyperedge e incident with the current vertex u. Then the surfer picks a destination vertex v within the chosen hyperedge satisfying the following $u, v \in e$. The

[2]By simple graphs, we mean graphs (not hypergraphs) with edges that are unique pair of vertices. Not to be confused with simple vs. multigraph

random walk in hypergraph is said to be more general since the random walk in a normal graph is a special case where there is only a single destination vertex v associated with a given normal edge incident with u where in a hypergraph we can have more vertices to choose from. The hypergraph random walk process can be defined as a Markov chain where the vertex set is the state set of the chain similar to a normal graph. At each time step t the surfer moves in the incident hyperedge to another vertex.

In this paper, we try to seek a general definition of a random walk in a weighted hypergraph where not only hyperedges have weights, but vertices as well. In such a case, the random walk process is extended to leverage both hyperedges' and vertices' weights. We define the vertex's weight across all incident hyperedges to be a feature vector

$$\vec{v}_w = \{w(v_{e1}), w(v_{e2}), ..., w(v_{d(v)})\} \tag{8}$$

Where we have a different vertex's weight for every hyperedge e that contain vertex v. We describe the proposed random walk process as the following. Starting from a vertex u, the surfer chooses a hyperedge e incident with u proportional to the hyperedge weight $w(e)$. Then, the surfer, also chooses a vertex v proportional to the vertex weight within the hyperedge where we consider the weight in the current hyperedge only. Let us define a weighted hypergraph incident matrix $H_w \in \mathbb{R}^{|V|\times|E|}$ where we have the following:

$$h_w(v,e) = \begin{cases} w(v_e) & \text{if } v \in e \\ 0 & \text{if } v \notin e \end{cases} \tag{9}$$

Therefore, we redefine the hyperedge degree to be as follows:

$$\delta(e_w) = \sum_{v\in V} h_w(v,e) \tag{10}$$

We can now calculate the transition matrix P as follows:

$$P(u,v) = \sum_{e\in E} w(e) \frac{h(u,e)}{\sum_{\hat{e}\in\mathcal{E}(u)} w(\hat{e})} \frac{h_w(v,e)}{\sum_{\hat{v}\in e} h_w(\hat{v},e)} \tag{11}$$

Or in matrix notation:

$$P = D_v^{-1} H W_e D_{ve}^{-1} H_w^T$$

Where $h_w(v,e)$ is the weight of the destination vertex v in hyperedge e. D_v is the diagonal matrix of the *weighted* degree of vertices as in formula 2. W_e is the diagonal matrix of the hyperedge weights. D_{ve} is the diagonal matrix for weighted degree of hyperedges as in formula 10. Note that the transition matrix P is *stochastic* where we have every row sums to 1.

After calculating the transition matrix P, we now explain the stationary distribution π of a random walk. The stationary distribution can be calculated by starting with initial column vector $\vec{v}_0 \in \mathbb{R}^{|V|\times 1}$ with equal probabilities $1/|V|$ summing to 1. We first multiply the transition matrix P^T(where P^T is a column stochastic matrix for clarity) by the initial column vector $\vec{v}_0$ yielding $\vec{v}_1 = P^T\vec{v}_0$. Then, we iterate until the vector $\vec{v}$ stops changing. The reason of multiplying the probability distribution vector $\vec{v}$ by the transition matrix P^T gives us the next step distribution $\vec{x} = P^T\vec{v}$ can be explained as follows. Let x_i be the probability of being at the current vertex i. Then we have the following: $x_i = \sum_j p_{ij}v_j$ where v_j being the probability of the surfer being at node j previously, and p_{ij} is the probability of moving from j to i.

The probability distribution vector $\vec{v}$ stops changing after n steps if the random walk is *ergodic*. A random walk is ergodic when the following conditions are met: 1) the chain is *irreducible*, for any two states $s_i, s_j \in \mathcal{M}$ they must satisfy $P(s_i, s_j) > 0$. Also, 2) the chain is *aperiodic*, where the greatest common divisor of every state $\{t : P_t(s_i, s_i) > 0\}$ is 1. To guarantee irreducibility and aperiodicity, we use the PageRank algorithm [6]. The algorithm uses the idea of *teleporting* which will restart the random walk process making it useful for the previous conditions. The teleporting is depicted with a small probability called the *damping factor* α. It also makes sure to make the graph irreducible since the random walker always has the probability of teleporting to any other node.

$$\vec{v}_{(i+1)} = \alpha P^T\vec{v}_{(i)} + (1-\alpha)\vec{e}/n \tag{12}$$

The damping factor α is set to 0.85. n is the number of nodes in the graph. $\vec{e} \in \mathbb{R}^{n\times 1}$ is a vector of all elements being 1. $\alpha P^T\vec{v}$ means that the random walker will choose to go with one of the incident hyperedges. $(1-\alpha)\vec{e}/n$ represents a vector of an introductory probabilities with each entry being $(1-\alpha)/n$ to teleport the random walk to a new node.

4.4 Extracting Keyphrases

We tag keywords with their Part of Speech (POS) tags. Then, we extract keyphrases that are noun phrases since it has been shown that most keyphrases annotated by human happen to be noun phrases[12][18][25]. We look for patterns as adj+nouns or all nouns and filter out the rest. Then, we have a candidate list of keyphrases based on the syntactic filtering that need to be ranked. A keyphrase ph is modeled as a collection of keywords k as $ph = \{k_1, k_2, ..., k_n\}$. To rank a keyphrase, most approaches aggregate the ranks of the keywords as follows:

$$R(ph) = \sum_{k_i\in ph} R(k_i) \tag{13}$$

However, such approach will be biased towards longer phrases. To overcome such bias, we normalized based on the length of the keyphrase as follows:

$$R(ph) = \frac{\sum_{k_i\in ph} R(k_i)}{n} \tag{14}$$

Where n is the keyphrase length. Moreover, we removed phrases that cross over syntactic boundaries as they cannot be a comprehensible keyphrase. We also removed any keyphrase that appears less than f times. We experimented with different values of f and found out that $f = 5$ shows the best keyphrases in our data. Next we will describe the experimental design in depth and all comparisons.

5. EXPERIMENT

This section explains the experimental setup for the hypergraph ranking framework HG-Rank. The effectiveness of our approach is demonstrated by conducting several experiments comparing our method to different baselines. First, the data sets used in this experiment are explained thoroughly. Second, the necessary preprocessing steps are illustrated in detail. Third, the experimental setup is laid out. Fourth, the experimental results and discussion of results are discussed and examined.

5.1 Data Sets

We used two different data sets that contain only short text documents. The characteristics of the two data sets are explained as the following:

- **Twitter.** We collected a corpus of tweets which contains 80,231 tweet posts. We collected tweets in the time frame from April 1, 2013 to April 30, 2013. We filter out all non-Latin characters tweets. Afterwards, we deleted any non-English tweets by classifying a tweet to be non-English if there is less than 5 English words. Moreover, we discarded any tweet with less than 3 words as it does not show any topic relevance. Moreover, the corpus contains 19,613 hashtags in total.

- **Opinosis.** We used a public short reviews data set called Opinosis[3] collected by Ganesan et al.[10]. The data set contains short reviews, a sentence long, about products collected from TripAdvisor, Amazon, and Edmunds. The data set contains 51 topics about a number of different products. For each topic, there is approximately a 100 short review snippet. A golden summary for each topic is created to summarize the reviews. There are 5 different golden summaries for each topic created by human workers from Amazon Mechanical Turk (MTurk)[4]. We randomly used 3 different topics to quantitatively test our algorithm with other baselines which are Windows7 features, iPod video, and Amazon's kindle price.

5.2 Preprocessing

Preprocessing is an essential step in text mining tasks in general. In extracting keyphrases, the preprocessing is needed to measure the salient scores accurately. The amount of preprocessing differs significantly depending on the genre of the corpus. In a social microblogging environment as in Twitter, the preprocessing step is of a vital importance. The challenge with colloquial textual content is an enormous obstacle in performing keyphrase extraction with Twitter posts. For instance, tweets can have misspelled words, strange capitalization, and wrong punctuations. For more detail we refer the reader to Eisenstein's survey on languages in social media [8]. Therefore, we did in an extensive preprocessing to tweets.

We first removed any URL links from tweets since we are focusing on the textual content. Moreover, we also removed emoticons and smileys since they do not have any topical relevance. Also, Twitter's special characters and usernames were removed as in the preceding hashtag sign # and usernames with @username. Tweets that start with the @username are generally considered replies and have a conversational nature more than topical nature. Therefore, we have removed any tweet that starts with @username to focus on topical tweets only. Another challenge is the usage of Internet phrasal abbreviation such as LOL (laugh out loud), ikr (I know right). We leverage the Internet Slang[5] dictionary in an effort to transform the text to standard English. All the aforementioned techniques can help improve the accuracy of the POS tagger.

Syntactical tagging, as in POS, for conversational content found on tweets can be very difficult. Most standard taggers fail to correctly tag colloquial text. For instance, the misuse of capitalization can make the tagger incorrectly tag nouns or verbs as a proper noun simply because the token is out of the vocabulary OOV. To tackle such difficulties, we have leveraged a state-of-the-art POS tagger[6] designed specifically for tweets [21]. The tagger designed at Carnegie Melon University is capable of accurately tagging tokens in a noisy genre as in Twitter. Moreover, the tagger is capable of identifying tags regardless of the capitalization misuse or the strange orthography of text, for example repeating letters for emphasis as in soooo. After tagging the tweets, we focused on selecting nouns and adjectives only since they are the base for noun phrases. We finally removed stopwords and stemmed the tokens.

The final step of preprocessing was to remove all stopwords from tweets since they do not have any topical influence. Punctuations were removed as well. Moreover, all capitalized tokens were converted to lower case. We lastly stemmed the tokens to get an accurate feature measure of words. We used the Porter stemmer[7] to stem our corpus.

For the Opinosis data set, we removed stopwords, punctuations, and stemmed the text. We also convert the tokens to lower case.

5.3 Experimental Setup

In this section, we will describe the experimental setup that was used for both data sets to compare our model with other baselines. First, we will describe the setup used for the Twitter data set. Then, we will explain the setup for the Opinosis review data.

Since there is no apparent golden labels to test against with tweets, we designed an empirical experiment to test keyphrase extraction in tweets. The experiment can be designed into different steps 1) Identify major topics in documents (tweets). 2) Test if any top ranked phrase represents a major topic in documents. The intuition behind the approach is the fact that phrases are descriptive of a document if they explain an important topic within that document. Given that keyphrases describe the major topics in a document, we will leverage a statistical topic model known as latent Dirichlet allocation (LDA)[4] to first extract the main topics in documents using a new twitter representation that improves the topic model with short documents. Second, we use search engines to search these topics to generate gold-label phrases. Finally, we test and compare all the ranking baselines by using the search results from the major topics in Twitter and golden summaries in Opinosis separately.

LDA is a generative statistical model that helps finding a set of unobserved groups using some observed sets. When applied to text, the observed sets are words in documents where the unobserved groups (latent) are topics of co-occurring words. By finding the mixture of topics using statistical inference as in Gibbes sampling, we get two posterior distribution $P(w|k)$ the probability of words under each topic, and $P(k|d)$ the probability of topics under each document. We first start by assigning each word to a K topic. Then, for each word w in each document d, we resample the

[3]http://kavita-ganesan.com/opinosis-opinion-dataset
[4]www.mturk.com
[5]http://www.noslang.com/dictionary/full/
[6]http://www.ark.cs.cmu.edu/TweetNLP/
[7]http://tartarus.org/martin/PorterStemmer/

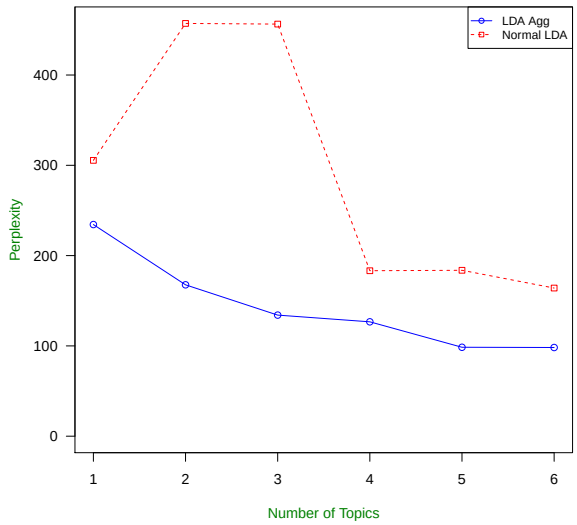


Figure 1: Comparison between LDA with Short Documents Compared to Aggregated Documents (↓)

probability distribution as follows:

$$P(w_i|d) = \sum_{s=1}^{|K|} P(w_i|k_s)P(k_s|d) \quad (15)$$

Where $P(w_i|k_s)$ is the probability of the word w_i being assigned topic k_s from all documents. $P(k_s|d)$ is the probability of words in the document d that are assigned to topic k_s. However, when applied to short length documents as in tweets, a new challenge arises since there is not enough observed sets (words) in each document to infer latent topics. Therefore, we remodel the documents structure to improve LDA in our data and get meaningful topics. Given a collection of tweets $\mathcal{T}_i = \{\tau_1, \tau_2, ..., \tau_n\}$, there is an abundant number of hashtags $\mathcal{H}_i = \{h_1, h_2, ..., h_n\}$ appearing in tweets. Instead of treating each tweet as a document, we aggregate tweets using hashtags to form a large virtual document for each hashtag $d_h = \{\tau_{1h} + \tau_{2h} + ... + \tau_{nh}\}$ where each d_h is a concatenation of tweets. Therefore, the documents set will be defined as $D_h = \{d_{h1}, d_{h2}, ..., d_{hn}\}$ containing all words from a large group of tweets for each document. After enhancing the document representation, we can apply LDA to learn topics and their posterior ranking more efficiently. In Figure 1, we show a significant improvement in perplexity for the two LDA approaches with short documents and aggregated documents. In Table 1, we show the top 10 ranked word for $|K| = 5$. Similar approaches for improving LDA in short text documents are found in [19].

To test the hypergraph ranking, we would need to have a reference set that summarize topics within each d_h document. The idea is to use a search engine by using the top words, from LDA, for each topic as a query. We used Google to generate the result snippets by setting the search at the same duration as the tweets which is April, 2013. Once the search snippets (top 50 snippet) for each topic is collected, we store them. Then we assign each document to its major topics only. For instance, any topic that is higher than $P(k|d) = 0.5$ is considered a major topic and is then chosen. Those search results collected from the major topics are considered references. We, then, search for keyphrases

Table 1: Top 10 Words of Five Topics

Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
super	health	obama	gun	social
bowl	care	barack	control	media
jar	law	presid	peopl	market
utc	congress	agenda	stricter	dimens
box	favor	buchanan	grip	roi
fuse	news	cien	sens	twitter
look	work	con	great	amp
good	alli	cumpl	america	use
beyonc	amp	mandato	anti	current
black	job	congress	common	post

in those topical references. If a keyphrase is included in a snippet of any major topic, we consider it a hit, otherwise it is a miss.

For the Opinosis review data, we compared the extracted keyphrases from short review documents with the golden summaries provided with the data set. We consider a keyphrase correct if it appears in any short golden summary. There is approximately 5 golden summaries for each topic. Due to the short length of documents, we only used bigrams for evaluation.

A number of different baseline algorithms are implemented and used to test the validity of the proposed approach:

- **tf-idf** each post is a document, and each topic collection is a reference corpus.
- **TextRank**[20] builds a graph of keywords with sliding window $w = 2$. Edge weights are the frequency of co-occurring relation.
- **TimedTextRank**[24] builds a graph of keywords similar to TextRank. The ranking is, however, multiplied by a time function over the destination node.
- **NGTS(normal graph with Time and Social)** A normal lexical graph similar to TextRank. However, the edge weights are the summation of temporal and social function over all documents that contain the pair of words u and v (instead of frequency of co-occurrence)
- **NE-Rank**[3] A normal lexical graph similar to TextRank. However, the ranking takes into account node and edge weights. Node weights are tf-idf and edge weights are the frequency of co-occurrence. It also considers node weights when the random walk teleports to a random node.

5.4 Experimental Results & Discussion

To compare all the baselines used in this experiment, we quantitatively measure their performance using a precision evaluation metric. We compare them to the golden labels defined in the previous section. Specifically, we consider the keyphrase to be correct if it appears in the golden set. Precision helps identify the accuracy of the extracted results. Since we are evaluating a ranking system, we measure precision at the top 10, 15, and 20 ranked keyphrases. Precision is measured as follows:

$$Precision = \frac{K_{correct}}{K_{extracted}} \quad (16)$$

Table 2: Keyphrase Extraction Experimental Results for Twitter using Precision

	P@10	P@15	P@20
tf-idf	0.28	0.26	0.24
TextRank	0.52	0.39	0.28
TimedTextRank	0.54	0.424	0.32
NGTS	0.48	0.40	0.32
NE-Rank	0.56	0.426	0.32
HG-RANK	0.64	0.49	0.40

Table 3: Keyphrase Extraction Experimental Results for Opinosis using Precision

	P@10	P@15	P@20
tf-idf	0.36	0.28	0.28
TextRank	0.53	0.42	0.35
NE-Rank	0.60	0.46	0.36
HG-RANK	0.66	0.57	0.46

Where $K_{correct}$ is the number of correctly extracted keyphrase, and $K_{extracted}$ is the total number of extracted keyphrase. In the following, we will discuss the experimental results with balancing social and temporal attributes and how it affect the ranking. Then, we will describe the improvements HG-RANK has over other approaches.

However, precision only considers how many correctly extracted keyphrases within the result regardless of the order within the top list of extracted keyphrases. Therefore, we also measure the Binary Preference Measure Bpref [7]. Bpref will penalize the system if incorrect keyphrases ranked higher than correct keyphrases. Bpref is measured as the following:

$$Bpref = \frac{1}{R}\sum_{r\in R} 1 - \frac{|\text{n } \textit{ranked higher than} \text{ r}|}{R} \qquad (17)$$

where R is the number of correct keywords within extracted keywords in a method, and where r is a correct keyword and n is incorrect keyword.

To examine the effect of social and temporal attributes when combined into the hypergraph ranking scheme, we experimented with different values for λ in formula 6. By varying the value of λ, we can analyze the tradeoff between the two attributes in precision. Figure 2 shows the different λ values experimented with. The best value for λ is 0.5 which means equal contribution of temporal and social features in our data. It is interesting to notice that when $\lambda = 1$, meaning only social attributes were taken into the ranking, the performance deteriorates considerably. It could mean that popular content is not necessary of a topical value to the corpus. However, more experiments are needed to widen our understanding of what the best features are for topical keyphrase extraction in a dynamic genre as Twitter. Next, we move on to describe the full evaluation of both data sets.

To evaluate the ranking performance for all baselines, we performed the evaluation measure for both data sets as the following. For Twitter, we first build the lexical hypergraph for each hashtag topic corpus $d_h = \{\tau_{1h} + \tau_{2h} + ... + \tau_{nh}\}$. We chose the top 5 frequent topical hashtags and performed keyphrase extraction separately. We measured the precision and Bpref for each topic. In table 2, we show the average precision from all topics. Table 4 shows the average Bpref for the Twitter data. In the Opinosis review data set, we build

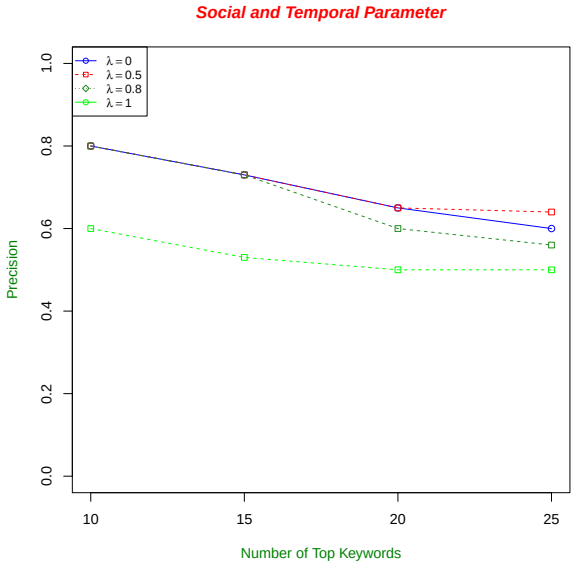


Figure 2: Different Effect between Temporal & Social Attributes

Table 4: Keyphrase Extraction Experimental Results for Twitter using Bpref

	Bpref@10	Bpref@15	Bpref@20
tf-idf	0.66	0.42	0.40
TextRank	0.744	0.74	0.74
TimedTextRank	0.748	0.67	0.67
NGTS	0.742	0.71	0.71
NE-Rank	0.76	0.75	0.75
HG-RANK	0.82	0.82	0.82

the lexical hypergraph for each topic. Since there is no meta data with the reviews as time or social features, we regard the hyperedge weight to be 1 for all short documents to test the hypergraph ranking only. We chose 3 topics mentioned early to quantitatively measure the improvements. We show the average precision of three topics in table 3, and average Bpref in table 5.

In the Twitter data, the proposed hypergraph-based approach HG-RANK out performed all other baselines. Specifically, HG-RANK improved the results in the top 10 results over closest baseline, NE-Rank, by 14% and 7% improvements using precision and Bpref, respectively. The improvement shows the importance of modeling the high-order co-occurring relationship using a lexical hypergraph compared to modeling just a pair of words for graph edges. Moreover, the temporally-aware ranking HG-RANK showed improvement over other temporal-aware approaches as in TimedTextRank[24] and NGTS. Similar improvements are demonstrated for the top 15 and top 20 keyphrases.

For the Opinosis data, HG-RANK showed improvement over all baselines as well. Improvements in the top 10 over the second best baseline, NE-Rank, were 10% in precision and 14% in Bpref. Moreover, similar improvements were found in the top 15 and top 20 keyphrases. Even though no hyperedge weights were used for this data set as in temporal and social attributes, the hypergraph model has shown to increase both precision and Bpref scores which shows the robustness of the proposed model in modeling high-order co-occurrence relation between words.

Table 5: Keyphrase Extraction Experimental Results for Opinosis using Bpref

	Bpref@10	Bpref@15	Bpref@20
tf-idf	0.61	0.61	0.61
TextRank	0.88	0.78	0.78
NE-Rank	0.82	0.80	0.80
HG-RANK	0.94	0.82	0.82

6. CONCLUSION

In this paper, we have proposed a hypergraph-based ranking algorithm suitable for short text documents in social media genre. We modeled distinct keywords as vertices and their short documents as hyperedges in a lexical hypergraph. Moreover, we jointly modeled temporal and social features in the hypergraph to adapt keyphrase extraction with the dynamic nature of social media. Additionally, we supplemented the hypergraph with discriminative weights in the vertices to enhance the random walk approach. Then we proposed a new probabilistic random walk that considers both vertices and hyperedges weights over hypergraph. We have leveraged a state-of-the-art POS tagger for Twitter data to capture syntactic tags accurately from the noisy text. We demonstrated the effectiveness of our hypergraph approach over two data sets which showed promising results.

In the future work, we plan to extend the approach to a streaming algorithm where the hypergraph can be updated periodically.

7. REFERENCES

[1] C. Avin, Y. Lando, and Z. Lotker. Radio cover time in hyper-graphs. In *Proceedings of the 6th International Workshop on Foundations of Mobile Computing*, DIALM-POMC '10, pages 3–12, New York, NY, USA, 2010. ACM.

[2] A. Bellaachia and M. Al-Dhelaan. Learning from twitter hashtags: Leveraging proximate tags to enhance graph-based keyphrase extraction. In *Proceedings of the 2012 IEEE GreenCom*, pages 348–357, Washington, DC, USA, 2012. IEEE Computer Society.

[3] A. Bellaachia and M. Al-Dhelaan. Ne-rank: A novel graph-based keyphrase extraction in twitter. In *Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence*, WI-IAT '12, pages 372–379, Washington, DC, USA, 2012. IEEE Computer Society.

[4] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022, Mar. 2003.

[5] A. Bougouin, F. Boudin, and B. Daille. Topicrank: Graph-based topic ranking for keyphrase extraction. In *Proceedings of the Sixth IJCNLP*, pages 543–551, Nagoya, Japan, October 2013. Asian Federation of Natural Language Processing.

[6] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the Seventh International Conference on World Wide Web 7*, pages 107–117. Elsevier Science Publishers B. V., 1998.

[7] C. Buckley and E. M. Voorhees. Retrieval evaluation with incomplete information. In *Proceedings of the 27th Annual International ACM SIGIR*, pages 25–32, New York, NY, USA, 2004. ACM.

[8] J. Eisenstein. What to do about bad language on the internet. In *Proceedings of the 2013 Conference of the NAACL*, pages 359–369, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[9] G. Erkan and D. R. Radev. Lexrank: graph-based lexical centrality as salience in summarization. *Journal of Artificial Intelligence Research*, 22(1):457–479, Dec. 2004.

[10] K. Ganesan, C. Zhai, and J. Han. Opinosis: a graph-based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd COLING*, pages 340–348, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[11] Y. Gao, J. Liu, and P. Ma. The hot keyphrase extraction based on tf*pdf. In *The 2011 IEEE 10th TrustCom*, pages 1524–1528, 2011.

[12] A. Hulth. Improved automatic keyword extraction given more linguistic knowledge. In M. Collins and M. Steedman, editors, *Proceedings of the 2003 EMNLP*, pages 216–223, 2003.

[13] L. Jabeur, L. Tamine, and M. Boughanem. Featured tweet search: Modeling time and social influence for microblog retrieval. In *Proceedings of the 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence*, volume 1, pages 166–173, 2012.

[14] D. Li and S. Li. Hypergraph-based inductive learning for generating implicit key phrases. In *Proceedings of the 20th WWW*, pages 77–78, New York, NY, USA, 2011. ACM.

[15] D. Li, S. Li, W. Li, W. Wang, and W. Qu. A semi-supervised key phrase extraction approach: Learning from title phrases through a document semantic network. In *Proceedings of the ACL*, pages 296–300, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[16] X. Li, B. Liu, and P. Yu. Time sensitive ranking with application to publication search. In *The Eighth IEEE ICDM*, pages 893–898, 2008.

[17] X. Liu, Y. Li, F. Wei, and M. Zhou. Graph-based multi-tweet summarization using social signals. In *Proceedings of COLING*, pages 1699–1714, Mumbai, India, December 2012. The COLING 2012 Organizing Committee.

[18] Z. Liu, W. Huang, Y. Zheng, and M. Sun. Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 EMNLP*, pages 366–376. Association for Computational Linguistics, October 2010.

[19] R. Mehrotra, S. Sanner, W. Buntine, and L. Xie. Improving lda topic models for microblogs via tweet pooling and automatic labeling. In *Proceedings of the 36th international ACM SIGIR*, pages 889–892, New York, NY, USA, 2013. ACM.

[20] R. Mihalcea and P. Tarau. Textrank: Bringing order into texts. In D. Lin and D. Wu, editors, *Proceedings of the 2004 EMNLP*, pages 404–411, Barcelona, Spain, July. Association for Computational Linguistics.

[21] O. Owoputi, B. O'Connor, C. Dyer, K. Gimpel, N. Schneider, and N. A. Smith. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of the 2013 Conference of the NAACL*, pages 380–390, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[22] Z. Ren, S. Liang, E. Meij, and M. de Rijke. Personalized time-aware tweets summarization. In *Proceedings of the 36th International ACM SIGIR*, pages 513–522, New York, NY, USA, 2013. ACM.

[23] R. Sipos, A. Swaminathan, P. Shivaswamy, and T. Joachims. Temporal corpus summarization using submodular word coverage. In *Proceedings of the 21st ACM CIKM*, pages 754–763, New York, NY, USA, 2012. ACM.

[24] X. Wan. Timedtextrank: adding the temporal dimension to multi-document summarization. In *Proceedings of the 30th annual international ACM SIGIR*, pages 867–868, New York, NY, USA, 2007. ACM.

[25] X. Wan and J. Xiao. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the 23rd National Conference on Artificial intelligence - Volume 2*, pages 855–860. AAAI Press, 2008.

[26] W. Wang, F. Wei, W. Li, and S. Li. Hypersum: hypergraph based semi-supervised sentence ranking for query-oriented summarization. In *Proceedings of the 18th ACM CIKM*, pages 1855–1858, New York, NY, USA, 2009. ACM.

[27] P. S. Yu, X. Li, and B. Liu. Adding the temporal dimension to search " a case study in publication search. In *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 543–549, Washington, DC, USA, 2005. IEEE Computer Society.

[28] X. Zhao, J. Jiang, J. He, Y. Song, P. Achanauparp, E.-P. Lim, and X. Li. Topical keyphrase extraction from twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 379–388, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[29] D. Zhou, J. Huang, and B. Scholkopf. Learning with hypergraphs: Clustering, classification, and embedding. *NIPS*, 19:1601, 2007.

Section III:

Poster Abstracts

Sentiment Analysis of Wimbledon Tweets

Priyanka Sinha
Tata Consultancy Services Limited
Ecospace 1B, New Town, Rajarhat
Kolkata 700156, India
priyanka27.s@tcs.com

Anirban Dutta Choudhury
Tata Consultancy Services Limited
Ecospace 1B, New Town, Rajarhat
Kolkata 700156, India
anirban.duttachoudhury@tcs.com

Amit Kumar Agrawal
Tata Consultancy Services Limited
Ecospace 1B, New Town, Rajarhat
Kolkata 700156, India
amitk.agrawal@tcs.com

ABSTRACT
Annotating videos in the absence of textual metadata is a major challenge as it involves complex image and video analytics, which is often error prone. However, if the video is a live coverage of an event, time correlated textual feed about the same event can act as a valuable source of aid for such annotation. Popular real time microblog streams like Twitter feeds can be an ideal source of such textual information. In this paper we explore the possibility of such correlation with the sentiment analysis of a set of tweets of the Roger Federer and Novak "Nole" Djokovic semi finals match at Wimbledon 2012.

Categories and Subject Descriptors
I.2.7 [**Natural Language Processing**]: Language Parsing and Understanding; H.1.2 [**User/Machine Systems**]: Human Information Processing

General Terms
Experimentation

Keywords
Twitter, Wimbeldon, Sentiment Analysis, TV

1. INTRODUCTION
Sentiment analysis or opinion mining is an automatic analysis of unstructured text to determine the sentiment expressed in the text such as the polarity of a sentence as either positive or negative.

One way to understand the sentiment of people viewing or experiencing the event is to analyze the video feed from TV or web hosting sites like Youtube. It is a challenging computationally hard problem especially without any helping text. Sentiment annotation in videos at finer granularity is not a much explored area. Sentiment annotations of a live video can be leveraged to enable targeted advertisement. However, there is problem with annotation quality due to the fact that manual video annotation is tedious and time consuming process whereas automated supervised video annotation is very limited in its coverage (i.e. incomplete and sometimes wrong). [7] is an example of how human crowdsourcing is one of the possible ways to annotate videos.

Based on our experiments detailed in "Our Approach" section, we observe that there exists a correlation between sentiment analysis on tweets and live coverage of video in real time of a popular event, i.e., the Wimbledon semi final match between Roger Federer and Novak Djokovic. This correlation is observed on this particular event and the confirmation of generality of the phenomenon is part of our future work. We have been successful in doing text analysis on microposts for sentiment analysis of a live event with respect to participants in that event in real time which has not been done before. We are proposing a novel approach for automatic sentiment annotation of live coverage of videos related to events affecting mass at large such as politics, natural calamities, sports etc. For a widely well known event with a large number of stakeholders, it is generally seen that the traffic on Twitter is huge with lots of people tweeting about it.

2. RELATED WORK
[8, 6] have demonstrated that twitter based sentiment analysis can be used for closely predicting political election results. However the approach is limited in temporal correlation because the political event (i.e., gold standard) is covered using various news flashes. It is not as fine grained and accurate as capturing the video of the unfolding of political events as they appear on either TV or web.

[5] is similar to our work in that they do discover named entities in tweets and micro-events for live events, but it is a different text mining task than sentiment analysis of the event.

3. OUR APPROACH
Tweets [2] have a maximum length of 140 characters. "Come on Federer! #Wimbledon" is an example tweet. RT is an acronym for retweet. @ is used to mention a twitter user name. # is used to represent a hashtag. http://bit.ly/9K4n9p is a shortened URL linking to external content.

Published as part of the #Microposts2014 Workshop proceedings, available online as CEUR Vol-1141 (http://ceur-ws.org/Vol-1141)

#Microposts2014, April 7th, 2014, Seoul, Korea.

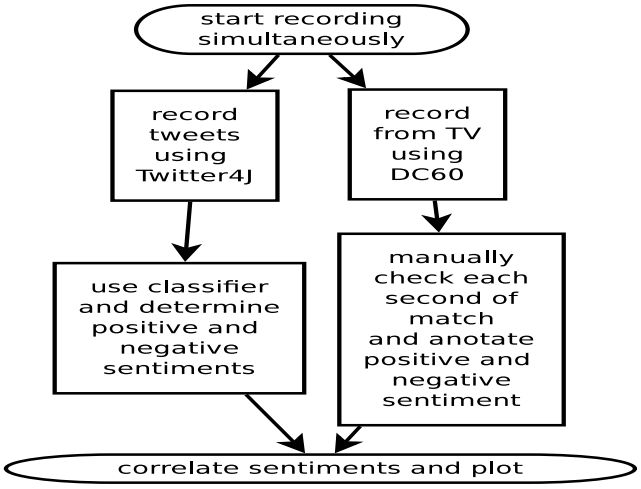


Figure 1: Flowchart for analyzing tweets and correlating results with TV video

Figure 1 depicts the steps we take to find the correlation between manually annotated sentiments of video and tweets. We set up one linux desktop to capture tweets using [4, 3]. Since Twitter uses OAuth 2.0 as authentication to connect to its API, we created an application and generated valid oauth token and secret online for use directly without the OAuth handshake. We grabbed the tweets for "Wimbeldon" keyword during the live telecast. In order to capture the TV video of the live telecast of wimbledon semi final match between "Roger Federer" and "Novak Djokovic", we tuned the Tata Sky set top box to Star Sports and attached a usb [1] to it connecting to a linux laptop. We used mencoder on the linux laptop with settings of aac for audio, h.264 for video and mp4 as the mux. The tweets and the video capture were started almost simultaneously thereby synchronizing the starting timestamps.

Three people manually annotated the video in two column format where the first column was "time in seconds since start of match" and second column was either "1" for positive sentiment for Roger or "2" for positive sentiment for Novak. Majority voting was taken to create the ground truth. We use supervised text classifiers such as Naive Bayes on tweets for sentiment polarity detection. We trained the classifier using part of the tweets which we manually annotated. Finally we used the sentiments derived from analysis on tweets and compared them to the ground truth. We found that the tweet sentiments were correlated with the video, and the time lag between video telecast and tweet was negligible.

4. CONCLUSION AND FUTURE WORK

When game sentiment is towards a particular player, advertisements endorsed by that player can be shown. We can also split the game into parts and get real time summarization of the game sentiment upto that point or within a time span.If intensity of sentiment is used then we can detect peaks of sentiments towards players as well and can tag best moments in the game as well.

Futuristic applications include allowing V-Commerce on live events like popular fashion shows where positive sentiment towards a participant of a video can inform the backend to adjust load towards possible increase in volume of incoming purchases.

Manual annotation to obtain the gold standard from video is a tedious task but gives an accurate understanding of the events sentiment. Video emotional analysis can be used to augment this process.

We understand that since this analysis has been done on one event, similar analysis on more popular real life events would help. Future work would also involve better techniques of sentiment analysis taking into account the short and noisy nature of tweets. Identification and treatment of languages other than english would help for certain events such as the Japanese tweets for Fukushima earthquake. Based on the correlation that we find between tweets and live events in the form of video, we are motivated enough to create a system where automatic annotation of live coverage of an event will take place using sentiment derived from tweets for the same event.

5. ACKNOWLEDGEMENTS

We would like to thank Chirabrata Bhaumik and Avik Ghose in helping us set up the video capture of the Wimbledon semi-finals which gave us our gold standard. We would also like to thank the workshop reviewers for their comments.

6. REFERENCES

[1] Dc 60. http://easycap.blogspot.in/p/easycap-dc60.html.
[2] Twitter. https://dev.twitter.com/docs/platform-objects/tweets.
[3] Twitter streaming api. https://dev.twitter.com/docs/api/1.1/post/statuses/filter.
[4] Twitter4j. http://twitter4j.org/en/index.html.
[5] S. Choudhury and J. G. Breslin. Extracting semantic entities and events from sports tweets. In *Making Sense of Microposts': Big Things Come in Small Packages: co-located with the 8th Extended Semantic Web Conference, ESWC2011*, Heraklion, Crete, May 2011.
[6] T. Rao and S. Srivastava. Analyzing stock market movements using twitter sentiment analysis. In *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*, ASONAM '12, pages 119–123, Washington, DC, USA, 2012. IEEE Computer Society.
[7] C. Vondrick, D. Patterson, and D. Ramanan. Efficiently scaling up crowdsourced video annotation. *International Journal of Computer Vision*, 101(1):184–204, 2013.
[8] H. Wang, D. Can, A. Kazemzadeh, F. Bar, and S. Narayanan. A system for real-time twitter sentiment analysis of 2012 u.s. presidential election cycle. In *Proceedings of the ACL 2012 System Demonstrations*, ACL '12, pages 115–120, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

Section IV:

Named Entity Extraction & Linking (NEEL) Challenge

edited by
Amparo Elizabeth Cano Basave, Giuseppe Rizzo & Andrea Varga
Matthew Rowe, Milan Stankovic & Aba-Sah Dadzie

Making Sense of Microposts (#Microposts2014) Named Entity Extraction & Linking Challenge

Amparo E. Cano
Knowledge Media Institute
The Open University, UK
amparo.cano@open.ac.uk

Giuseppe Rizzo
Università di Torino, Italy
EURECOM, France
giuseppe.rizzo@di.unito.it

Andrea Varga
The OAK Group
The University of Sheffield, UK
a.varga@dcs.shef.ac.uk

Matthew Rowe
School of Computing and Communications
Lancaster University, UK
m.rowe@lancaster.ac.uk

Milan Stankovic
Sépage, France
Université Paris-Sorbonne
milstan@gmail.com

Aba-Sah Dadzie
School of Computer Science
University of Birmingham, UK
a.dadzie@cs.bham.ac.uk

ABSTRACT

Microposts are small fragments of social media content and a popular medium for sharing facts, opinions and emotions. They comprise a wealth of data which is increasing exponentially, and which therefore presents new challenges for the information extraction community, among others. This paper describes the 'Making Sense of Microposts' (#Microposts2014) Workshop's Named Entity Extraction and Linking (NEEL) Challenge, held as part of the 2014 World Wide Web conference (WWW'14). The task of this challenge consists of the automatic extraction and linkage of entities appearing within English Microposts on Twitter. Participants were set the task of engineering a named entity extraction and DBpedia linkage system targeting a predefined taxonomy, to be run on the challenge data set, comprising a manually annotated training and a test corpus of Microposts. 43 research groups expressed intent to participate in the challenge, of which 24 signed the agreement required to be given a copy of the training and test datasets. 8 groups fulfilled all submission requirements, out of which 4 were accepted for the presentation at the workshop and a further 2 as posters. The submissions covered sequential and joint methods for approaching the named entity extraction and entity linking tasks. We describe the evaluation process and discuss the performance of the different approaches to the #Microposts2014 NEEL Challenge.

Keywords

Microposts, Named Entity, Evaluation, Extraction, Linking, Disambiguation, Challenge

1. INTRODUCTION

Since the first Making Sense of Microposts (#MSM2011) workshop at the Extended Semantic Web Conference in 2011 through to the most recent workshop in 2014 we have received over 80 submissions covering a wide range of topics related to mining information and (re-)using the knowledge content of Microposts. Microposts are short text messages published using minimal effort via social media platforms. They provide a publicly available wealth of data which has proven to be useful in different applications and contexts (e.g. music recommendation, social bots, emergency response situations). However, gleaning useful information from Micropost content presents various challenges, due, among others, to the inherent characteristics of this type of data:

i) the limited length of Microposts;
ii) the noisy lexical nature of Microposts, where terminology differs between users when referring to the same thing, and abbreviations are commonplace.

A commonly used approach for mining Microposts is the use of cues that are available in textual documents, providing contextual features to this content. One example of such a cue is the use of named entities (NE). Extracting named entities in Micropost content has proved to be a challenging task; this was the focus of the first challenge, in #MSM2013 [3]. A step further into the use of such cues is to be able not only to recognize and classify them but also to provide further information, in other words, disambiguating entities. This prompted the Named Entity Extraction and Linking (NEEL) Challenge, held as part of the *Making Sense of Microposts Workshop (#Microposts2014)* at the *2014 World Wide Web Conference (WWW'14)*.

The purpose of this challenge was to set up an open and competitive environment that would encourage participants to deliver novel or improved approaches to extract entities from Microposts and link them to their DBpedia counterpart resources (if defined). This report describes the #Microposts2014 NEEL Challenge, our collaborative annotation of a corpus of Microposts and our evaluation of the performance of each submission. We also describe the approaches taken in the participants' systems – which use both established and novel, alternative approaches to entity extraction and linking. We describe how well they performed and how system performance differed across approaches. The resulting body of work has implications for researchers interested in the task of information extraction from social media.

2. THE CHALLENGE

In this section we describe the goal of the challenge, the task set, and the process we followed to generate the corpus of Microposts. We conclude the section with the list of the accepted submissions.

Published as part of the #Microposts2014 Workshop proceedings, available online as CEUR Vol-1141 (http://ceur-ws.org/Vol-1141)

#Microposts2014, April 7th, 2014, Seoul, Korea.

2.1 The Task and Goal

The NEEL Challenge task required participants to build semi-automated systems in two stages:

(i) generally known as Named Entity Extraction (NEE) – in which participants were to extract entity mentions from a tweet; and

(ii) known as Named Entity Linking (NEL), in which each entity extracted is linked to an English DBpedia v3.9 resource.

For this task we considered the definition of an *entity* in the general sense of being, in which an object or a set of objects do not necessarily need to have a material existence, but which however must be characterized as an instance of a taxonomy class. To facilitate the creation of the *gold standard* (GS) we limited the entity types evaluated in this challenge by specifying the taxonomy to be used: the NERD ontology v0.5[1] [16]. To this we added a few concepts from the DBpedia taxonomy. The taxonomy was not considered as normative in the evaluation of the submissions, nor for the ranking. This is a deliberate choice, to increase the complexity of the task and to let participants perform taxonomy matching starting from the distribution of the entities in the GS. The list of classes in the taxonomy used is distributed with the released GS[2].

Beside the typical word-tokens found in a Micropost, new to this year's challenge we considered special social media markers as entity mentions as well. These Twitter markers are tokens introduced with a special symbol. We considered two such markers: *hashtags*, prefixed by #, denoting the topic of a Micropost (e.g. #*londonriots*, #*surreyriots*, #*osloexpl*), and *mentions* prefixed by @, referring to Twitter user names, which include entities such as organizations (e.g. @*bbcworldservice*) and celebrities (e.g. @*ChadMMurray*, @*AmyWinehouse*).

Participants were required to recognize these different entity types within a given Micropost, and to extract the corresponding entity link tuples. Consider the following example, taken from our annotated corpus:

Source (tweet text):
`RT @bbcworldservice police confirms bomb in Oslo #oslexp`

The 2^{nd} token (the mention @*bbcworldservice*) in this Micropost refers to the international broadcaster, the BBC World Service; the 7^{th} token refers to the location *Oslo*; while the 8^{th} token (the hashtag #*oslexp*) refers to the 2011 Norway terrorist attack. An entry to the challenge would be required to spot these tokens and display the result as a set of annotations, where each line corresponds to a tab-separated *entity mention*[3] and *entity link*[4]:

Correctly formatted result:
`bbcworldservice` ***dbpedia:BBC_World_Service***
`Oslo` ***dbpedia:Oslo***
`oslexp` ***dbpedia:2011_Norway_attacks***

We also consider the case where an entity is referenced in a tweet either as a noun or a noun phrase, if it:

a) belongs to one of the categories specified in the taxonomy;
b) is disambiguated by a DBpedia URI within the context of the tweet. Hence any (single word or phrase) entity without a disambiguation URI is disregarded;
c) subsumes other entities. The longest entity phrase within a Micropost, composed of multiple sequential entities and that can be disambiguated by a DBpedia URI, takes precedence over its component entities.

Consider the following examples:

1. **[*Natural History Museum at Tring*]**;
2. **[*News International chairman James Murdoch*]**`'s evidence to MPs on phone hacking`;
3. **[*Sony*]**`'s` **[*Android Honeycomb*]** `Tablet`

For the 3^{nd} case, even though they may appear to be a coherent phrases, since there are no DBpedia URIs for `[Sony's Android Honeycomb]` or `[Sony's Android Honeycomb Tablet]`, the entity phrase is split into what are the (valid) component entities highlighted above.

To encourage competition we solicited sponsorship for the winning submission. This was provided by the European project LinkedTV[5], who offered a prize of an iPad This generous sponsorship is testament to the growing interest in issues related to automatic approaches for gleaning information from (the very large amounts of) social media data.

2.2 Data Collection and Annotation

The challenge data set comprises 3,505 tweets extracted from a collection of over 18 million tweets. This collection, provided by the Redites project[6], covers event-annotated tweets collected for the period 15th July 2011 to 15th August 2011 (31 days). It extends over multiple notable events, including the death of Amy Winehouse, the London Riots and the Oslo bombing. Since the NEEL Challenge task is to automatically extract and link entities, we built our data set considering both event and non-event tweets. Event tweets are more likely to contain entities; non-event tweets therefore enable us to evaluate the performance of the system in avoiding false positives in the entity extraction phase.

Statistics describing the training and test sets are provided in Table 1. The dataset was split into training (70%) and test (30%) sets. The training set contains 2,340 tweets, with 41,037 tokens and 3,819 named entities; the test set contains 1,165 tweets, with 20,224 tokens and 1,458 named entities. The tweets are relatively

[1] `http://nerd.eurecom.fr/ontology/nerd-v0.5.n3`
[2] The NEEL Challenge GS available for download from: `http://ceur-ws.org/Vol-1141/microposts2014-neel_challenge_gs.zip`
[3] Note that the annotated result returns tokens without the social media markers (# and @) in the original Micropost.
[4] In this example "dbpedia:" refers to the namespace prefix of a DBpedia resource (see `http://dbpedia.org/resource`)
[5] `http://www.linkedtv.eu`
[6] `http://demeter.inf.ed.ac.uk/redites`

Table 1: General statistics of the training and test data sets:
***Posts* refers to the number of tweets in a data set; *Words* to the unique number of words; *Tokens* refers to the total number of words; *AvgTokens/Post* represents the average number of tokens per tweet; *NEs* denotes the unique number of NEs; *totalNEs* the total number of NEs; and *AvgNEs/Post* the average number of NEs per post. We computed AvgTokens/Post and AvgNEs/Post as the standard deviation from the mean (mean ± standard deviation).**

Dataset	Posts	Words/Tokens	AvgTokens/Post	NEs	totalNEs	AvgNEs/Post
train	2,340	12,758/41,037	17.54±5.70	1,862	3,819	3.26±3.37
test	1,165	6,858/20,224	17.36±5.59	834	1,458	2.50±2.94

long in both data sets; the average number of tokens per tweet is 17.54±5.70 in the training, and 17.36±5.59 in the test set. The average number of entities per tweet is also relatively high, at 3.26±3.37 for the training and 2.50±2.94 for the test dataset. The percentage of tweets without any valid entities is 32% (775 tweets) in the training, and 40% (469 tweets) in the test set. There is a fair bit of overlap of entities between the training and test data: 13.27% (316) of the named entities in the training data also occurs in the test dataset. With regard to the tokens in the original tweets with hashtag and mention social media markers, a total of 406 hashtags represented valid entities in the training, with 184 in the test set. The total number of valid entity mentions was 133 in the training, and 73 in the test data set.

The annotation of each Micropost in the training set gave all participants a common base from which to learn extraction patterns. In order to assess the performance of the submissions we used an underlying *gold standard* (GS), generated by 14 annotators, who had different backgrounds, including computer scientists, social scientists, social web experts, semantic web experts and linguists.

The annotation process comprised the following phases[7]

Phase 1. Unsupervised annotation of the corpus was performed, to extract candidate links that were used as input to the next stage. The candidates were extracted using the NERD framework [15].

Phase 2. The data set was divided into batches, with three different annotators to each batch. In this phase annotations were performed using CrowdFlower[8]. The annotators were asked to analyze the NERD links generated in phase 1 by adding or removing entity-annotations as required. The annotators were also asked to mark any ambiguous cases encountered.

Phase 3. In the final stage, consistency checking, three experts double-checked the annotations and generated the GS (for both the training and test sets). Three main tasks were carried out here: (1) cross-consistency check of entity types; (2) cross-consistency check of URIs; (3) resolution of ambiguous cases raised by the 14 annotators.

The complete data set, including a list of changes and the gold standard, is available for download[9] with the #Microposts2014 Workshop proceedings, accessible under the Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License[10].

[7]We aim to provide a more detailed explanation of the annotation process and the rest of the NEEL Challenge evaluation process in a separate publication.
[8]http://crowdflower.com

Table 2: Submissions accepted, ordered by submission number, with team affiliations and number of runs for each.

ID	Affiliation	Authors	Runs
13	UTwente	Habib, M. et al.	2
15	Max Planck	Amir, M. et al.	3
16	IIT Hyberabad	Bansal, R. et al.	1
18	Microsoft	Chang, M.	3
19	Net7-Spaziodati-UPisa	Scaiella, U. et al.	2
20	SAP	Dahlmeier, D. et al.	1

2.3 Challenge Submissions

The challenge attracted a lot of interest from research groups spread across the world. Initially, 43 groups expressed their intent to participate in the challenge; however only 8 completed submission. Each submission consisted of a short paper explaining the system approach, and up to three different test set annotations generated by running the system with different settings. After peer review, 4 submissions were accepted, and a further 2 as posters. The submission run with the best overall performance for each system was used in the rankings (see Table 4). The submissions accepted are listed in Table 2.

2.4 System Descriptions

We present next an analysis of the participants' systems for the Named Entity Extraction and Linking (NEEL) tasks. Except for submission 18, who treated the NEEL task as a joint task of Named Entity Extraction (NEE) and Named Entity Linking (NEL); all participants approached the NEEL task as two sequential sub-tasks (i.e. NEE first, followed by NEL). A summary of these approaches includes:

i) use of *external systems*;
ii) *main features* used;
iii) type of *strategy* used;

[9]http://ceur-ws.org/Vol-1141/microposts2014-neel_challenge_gs.zip
[10]Following the Twitter ToS we only provide tweet IDs and annotations for the training set; and tweet IDs for the test set.

iv) use of *external sources*.

Table 3 provides a detailed summary of the approaches used for both the NEE and NEL tasks.

The NEE task on Microposts is on its own challenging. One of the main *strategies* was to use off-the-shelf named entity recognition (NER) tools, improved through the use of extended gazetteers. System 18 approached the NEE task from scratch using a rule-based approach; all others made use of *external toolkits*. Some of these were Twitter-tuned and were applied for:

i) feature extraction, including the use of the TwitterNLP (2013) [13] and TwitterNLP (2011) [8] toolkits for POS tagging (systems 16, 20);

ii) entity extraction with TwiNER [10], Ritter's NER [14] and TAGME [5] (systems 13, 16, 19).

Other *external toolkits* which address NEE in longer newswire texts were also applied, including Stanford NER [6] and DBpedia Spotlight [11] (systems 15, 20).

Another common trend across these systems was the use of gazetteer-based, rule-matching approaches to improve the coverage of the off-the-shelf tools. System 13 applied simple regular expression rules to detect additional named entities not found by the NE extractor (such as numbers, and dates); systems 15 and 18 applied rules to find candidate entity mentions using a knowledge base (among others, Freebase [2]). Some systems also applied name normalization for feature extraction (systems 15, 18). This strategy was particularly useful for catering for entities originally appearing as hashtags or username mentions. For example, hashtags such as #BarackObama were normalized into a composite entity mention "Barack Obama"; and "@EmWatson" into "Emma Watson".

The NEL task involved in some cases the use of off-the-self tools, for finding candidate links for each entity mention and/or for deriving mention features (systems 13, 19, 20). A common trend across systems was the use of external knowledge sources including:

i) NER dictionaries (e.g. Google CrossWiki [17]);

ii) Knowledge Base Gazetteers (e.g. Yago [9], DBpedia [1]);

iii) Weighted lexicons (using e.g. Freebase [2], Wikipedia);

iv) other sources (e.g. Microsoft Web N-gram [19]).

A wide range of different *features* was investigated for the linking strategies. Some systems characterized an entity using Micropost-derived features with Knowledge base (KB)-derived features (systems 13, 15, 16, 19). Micropost-derived features include the use of lexical (e.g., N-grams, capitalization) and syntactical (e.g., POS) features, while KB-derived features included the use of URIs, anchor text and link-based probabilities (see Table 3). Additionally, features were extended by capturing jointly the local (within a Micropost) and global (within a knowledge base) contextual information of an entity, via graph-based features (such as entity semantic cohesiveness) (system 18). Further novel features included the use of Twitter account metadata for characterizing mentions and popularity-based statistical features for characterizing entities (systems 16, 18).

The classification *strategies* used for entity linking included supervised approaches (systems 13, 15, 16, 18, 19) existing off-the-shelf approaches enhanced with simple heuristics (e.g. the search+rules) (system 20).

3. EVALUATION OF CHALLENGE SUBMISSIONS

We describe next the evaluation measures used to assess the goodness of the submissions and conclude with the final challenge rankings, with submissions ordered according to the F_1 measure.

3.1 Evaluation Measures

We evaluate the goodness of a system S in terms of the performance of the system to both recognize and link an entity from a test set TS. Per each instance in TS, a system provides a set of pairs P of the form: entity mention (e), and link (l). A link is any valid DBpedia URI[11] that points to an existing resource (e.g. `http://dbpedia.org/resource/Barack_Obama`). The evaluation consists of comparing submission entry pairs against those in the gold standard GS. The measures used to evaluate each pair are precision P, recall R, and f-measure F_1. The evaluation is based on micro-averages.

First, a cleansing stage is performed over each submission, resolving where needed, the redirects. Then, to assess the correctness of the pairs provided by a system S, we perform an exact-match evaluation, in which a pair is correct only if both the entity mention and the link match the corresponding set in the GS. Pair order is also relevant. We define $(e,l)_S \in S$ as the set of pairs extracted by the system S, $(e,l)_{GS} \in GS$ denotes the set of pairs in the gold standard. We define the set of true positives TP, false positives FP, and false negatives FN for a given system as:

$$TP = \{(e,l)_S | (e,l)_{GS} \in (S \cap GS)\} \quad (1)$$

$$FP = \{(e,l)_S | (e,l)_{GS} \in S \wedge (e,l) \notin GS)\} \quad (2)$$

$$FN = \{(e,l)_S | (e,l)_{GS} \in GS \wedge (e,l) \notin S\} \quad (3)$$

Thus TP defines the set of relevant pairs in TS, in other words the set of pairs in TS that match corresponding ones in GS. FP is the set of irrelevant pairs in TS, in other words the pairs in TS that do not match the pairs in GS. FN is the set of false negatives denoting the pairs that are not recognised by TS, yet appear in GS. Since our evaluation is based on a micro-average analysis, we sum the individual true positives, false positives, and false negatives of each system across all Microposts. As we require an exact-match for pairs (e,l) we are looking for strict entity recognition and linking matches; each system has to link each entity e recognised to the correct resource l.

From this set of definitions, we define precision, recall, and f-measure as follows:

$$P = \frac{|TP|}{|TP \cup FP|} \quad (4)$$

[11]We consider all DBpedia v3.9 resources valid.

Table 3: Automated approaches used in #Microposts2014 NEEL Challenge

Named Entity Extraction (NEE)

	External System	Main Features	NE Extraction Strategy	Linguistic Knowledge
13 UTwente	TwiNER [10]	Regular Expression, Entity phrases, N-gram	TwiNER [10] + CRF	DB[a] Gazetteer [1], Wiki[b] Gazetteer
15 MaxPlanck	StanfordNER [6]	-	-	NER Dictionary
16 IIT Hyberabad	RitterNER [14],TwitterNLP(2011) [8]	Proper nouns sequence, N-grams	-	Wiki
18 Microsoft	-	N-grams, stop words removal, punctuation as tokens	Rule-based (candidate filter)	Wiki and Freebase [2] lexicons
19 Net7-Spaziodati-UPisa	TAGME [5]	Wiki anchor texts, N-grams	Collective agreement + Wiki stats	Wiki
20 SAP	DBSpotlight [11], TwitterNLP(2013) [13]	Unigram, POS, lower, title & upper case, stripped words, isNumber, word cluster, DBpedia	CRF	DB Gazetteer, Brown Clusters [18]

Named Entity Linking (NEL)

	External System	Main Features	Linking Strategy	Linguistic Knowledge
13 UTwente	Google Search[c]	N-grams, DB links, Wiki links, Capitalization	SVM	Wiki, DB, WordNet [12], Web N-Gram [19], Yago [9]
15 MaxPlanck	-	Prefix, POS, suffix, Twitter account metadata, normalized mentions, tri-grams	entity aggregate prior + prefix-tree data structure + DB match	Wiki, DB, Yago [9]
16 IIT Hyberabad	-	Wiki context-based measure, anchor text measure, entity popularity (in Twitter)	LambdaMART [20] (ranking/disambiguation)	Wiki Gazetteer, Google CrossWiki Dictionary [17]
18 Microsoft	-	N-grams, lower case, entity graph features (entity semantic cohesiveness), popularity-based statistical features (clicks and visiting information from the Web)	DCD-SSVM [4] + MART gradient boosting [7]	Wiki, Freebase [2]
19 Net7-Spaziodati-UPisa	TAGME [5]	Link probability, mention-link commonness	C4.5 (for taxonomy-filter)	Wiki, DB [1]
20 SAP	Search API (Wiki, DBSpotlight [11], Google)	Entity mention	Search+rules	Wiki, DB [1]

[a] DBpedia abbreviated to 'DB' [1]
[b] Wikipedia abbreviated to 'Wiki'
[c] Google Search, https://developers.google.com/web-search/docs

$$R = \frac{|TP|}{|TP \cup FN|} \quad (5)$$

$$F_1 = 2 * \frac{P * R}{P + R} \quad (6)$$

The evaluation framework used in the challenge is available at `https://github.com/giusepperizzo/neeleval`.

3.2 Evaluation Results

Table 4 reports the performance of participants' systems, using the best run for each. The ranking is based on the F_1.

Table 4: P, R, F_1 breakdown figures per submission.

Rank	System	Entry	P	R	F_1
1	18-2	Microsoft	77.10	64.20	70.06
2	13-2	UTwente	57.30	52.74	54.93
3	19-2	Net7-Spaziodati-UPisa	60.93	42.25	49.90
4	15-3	MaxPlanck	53.28	39.51	45.37
5	16-1	IIT Hyberabad	50.95	40.67	45.23
6	20-1	SAP	49.58	32.17	39.02

System 18 clearly outperformed other systems, with F_1 more than 15% higher than the next best system. System 18 differed from all other systems, by using a joint approach to the NEEL task. The others each divided the task into a sequential entity extraction and linking task. The approach in System 18 made use of features which capture jointly an entity's local and global contextual information, resulting in the best approach submitted to the #Microposts2014 NEEL Challenge.

4. CONCLUSIONS

The aim of the #Microposts2014 Named Entity Extraction & Linking Challenge was to foster an open initiative that would encourage participants to develop novel approaches for extracting and linking entity mentions appearing in Microposts. The NEEL task involved the extraction of entity mentions in Microposts and the linking of these entity mentions to DBpedia resources (where such exist).

Our motivation for hosting this challenge is the increased availability of third-party entity extraction and entity linking tools. Such tools have proven to be a good starting point for entity linking, even for Microposts. However, the evaluation results show that the NEEL task remains challenging when applied to social media content with its peculiarities, when compared to standard length text employing regular language.

As a result of this challenge, and the collaboration of annotators and participants, we also generated a manually annotated data set, which may be used in conjunction with the NEEL evaluation framework (`neeleval`). To the best of our knowledge this is the largest publicly available data set providing entity/resource annotations for Microposts. We hope that both the data set and the `neeleval` framework will facilitate the development of future approaches in this and other such tasks.

The results of this challenge highlighted the relevance of normalization and time-dependent features (such as popularity) for dealing with this type of progressively changing content. It also indicated that learning entity extraction and linking as a joint task may be beneficial for boosting performance in entity linking in Microposts.

We aim to continue to host additional challenges targeting more complex tasks, within the context of data mining of Microposts.

5. ACKNOWLEDGMENTS

The authors thank Nikolaos Aletras for helping to set-up the crowdflower experiments for annotating the gold standard data set. A special thank to Andrés García-Silva, Daniel Preoţiuc-Pietro, Ebrahim Bagheri, José M. Morales del Castillo, Irina Temnikova, Georgios Paltoglou, Pierpaolo Basile and Leon Derczynski, who took part in the annotation tasks. We also thank the participants who helped us improve the gold standard. We finally thank the LinkedTV project for supporting the challenge by sponsoring the prize for the winning submission.

6. REFERENCES

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, and Z. Ives. DBpedia: A Nucleus for a Web of Open Data. In *6^{th} International Semantic Web Conference (ISWC'07)*, 2007.

[2] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *ACM SIGMOD International Conference on Management of Data (SIGMOD'08)*, 2008.

[3] A. E. Cano Basave, A. Varga, M. Rowe, M. Stankovic, and A.-S. Dadzie. Making Sense of Microposts (#MSM2013) Concept Extraction Challenge. In *Making Sense of Microposts (#MSM2013) Concept Extraction Challenge*, 2013.

[4] M.-W. Chang and W.-T. Yih. Dual coordinate descent algorithms for efficient large margin structured prediction. *Transactions of the Association for Computational Linguistics*, 2013.

[5] P. Ferragina and U. Scaiella. Fast and accurate annotation of short texts with Wikipedia pages. *IEEE Software*, 29(1):70–75, 2012.

[6] J. R. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by Gibbs sampling. In *43^{rd} Annual Meeting on Association for Computational Linguistics (ACL'05)*, 2005.

[7] J. H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000.

[8] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. A. Smith. Part-of-speech tagging for Twitter: Annotation, features, and experiments. In *49^{th} Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011.

[9] J. Hoffart, F. M. Suchanek, K. Berberich, E. Lewis-Kelham, G. de Melo, and G. Weikum. YAGO2: Exploring and querying world knowledge in time, space, context, and many languages. In *20^{th} International Conference Companion on World Wide Web (WWW'11)*, 2011.

[10] C. Li, J. Weng, Q. He, Y. Yao, A. Datta, A. Sun, and B.-S. Lee. TwiNER: Named entity recognition in targeted Twitter stream. In *35^{th} International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '12)*, 2012.

[11] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer.

DBpedia spotlight: shedding light on the web of documents. In *7th International Conference on Semantic Systems (I-Semantics'11)*, 2011.

[12] G. A. Miller. Wordnet: A lexical database for English. *Commun. ACM*, 38(11):39–41, 1995.

[13] O. Owoputi, C. Dyer, K. Gimpel, N. Schneider, and N. A. Smith. Improved part-of-speech tagging for online conversational text with word clusters. In *NAACL*, 2013.

[14] A. Ritter, S. Clark, Mausam, and O. Etzioni. Named entity recognition in tweets: An experimental study. In *EMNLP*, 2011.

[15] G. Rizzo and R. Troncy. NERD: A framework for unifying named entity recognition and disambiguation extraction tools. In *13th Conference of the European Chapter of the Association for computational Linguistics (EACL'12)*, 2012.

[16] G. Rizzo, R. Troncy, S. Hellmann, and M. Bruemmer. NERD meets NIF: Lifting NLP extraction results to the Linked Data Cloud. In *Proceedings of the 5th International Workshop on Linked Data on the Web (LDOW'12)*, 2012.

[17] V. I. Spitkovsky and A. X. Chang. A cross-lingual dictionary for English Wikipedia concepts. In *LREC*, 2012.

[18] J. Turian, L. Ratinov, and Y. Bengio. Word representations: A simple and general method for semi-supervised learning. In *48th Annual Meeting of the Association for Computational Linguistics (ACL'10)*, 2010.

[19] K. Wang, C. Thrasher, E. Viegas, X. Li, and B.-j. P. Hsu. An overview of Microsoft Web N-gram corpus and applications. In *NAACL HLT 2010 Demonstration Session (HLT-DEMO'10)*, 2010.

[20] Q. Wu, C. J. Burges, K. M. Svore, and J. Gao. Adapting boosting for information retrieval measures. *Information Retrieval*, 13(3):254–270, 2010.

Section IVa:

NEEL Challenge Submissions I

E2E: An End-to-End Entity Linking System for Short and Noisy Text

Ming-Wei Chang Bo-June Hsu Hao Ma Ricky Loynd Kuansan Wang
Microsoft Research
Redmond, WA
{minchang|paulhsu|haoma|riloynd|kuansanw}@microsoft.com

ABSTRACT

We present E2E, an end-to-end entity linking system that is designed for short and noisy text found in microblogs and text messages. Mining and extracting entities from short text is an essential step for many content analysis applications. By jointly optimizing entity recognition and disambiguation as a single task, our system can process short and noisy text robustly.

Keywords

Information Extraction, Social Media, Entity Linking

1. INTRODUCTION

In this paper, we describe our entity linking system called E2E for the #Microposts2014 NEEL Challenge [1]. Our system focuses on the task of extracting and linking entities from short and noisy text given entity databases like Wikipedia or Freebase. An entity linking system usually needs to perform two key functions: mention recognition and entity disambiguation. In mention recognition the system identifies each mention (surface form) of an entity in the text. In entity disambiguation, the system maps mentions to canonical entities. E2E has been carefully designed to treats entity recognition and disambiguation as a single task.

2. THE ARCHITECTURE OF E2E

When a short message is received, E2E processes the message in four stages: Text Normalization, Candidate Generation and Joint Recognition-and-Disambiguation, and Overlap Resolution.

Text Normalization.

In this stage, a short message is normalized and tokenized. For tweets, the retweet symbols and some other special symbols are removed. Punctuation symbols are represented as separate tokens in general.

Published as part of the #Microposts2014 Workshop proceedings, available online as CEUR Vol-1141 (http://ceur-ws.org/Vol-1141)

#Microposts2014, April 7th, 2014, Seoul, Korea.

Candidate Generation.

The next step is to generate a list of surface form candidates that could potentially link to entities. E2E uses a lexicon to generate the candidate surface forms. A lexicon is a dictionary that maps a surface form to its possible entity set. For example, the word "giants" could refer to "New York Giants", or "San Francisco Giants", etc. Our lexicon is mainly composed by extracting information from Wikipedia and Freebase. The dictionary is constructed to support fuzzy mention matching based on edit distance. Note that we over-generate candidates at this stage, and no filtering is performed.

Joint Recognition and Disambiguation.

This stage is the key component of the E2E framework. Given a message, the goal here is to figure out the entity assignment of each candidate mention generated from previous stages. Note that a candidate mention may be rejected altogether (mapped to the null entity).

Our model is based on a supervised learning method. Given a message m and a candidate mention a, the entity assignment is generated from the ranking of all possible entities in the entity set $\mathcal{E}(a)$.

$$\arg\max_{e \in \{\mathcal{E}(a) \cap \emptyset\}} f(\Phi(m, a, e)), \quad (1)$$

where f is the function of the model, and Φ is a feature function over the input m, the mention a and the candidate output e. Note that it is very likely E2E rejects a candidate and does not link it to an entity (link a to $\emptyset$). The joint approach that recognizes and disambiguates entity mentions together is crucial for E2E to properly link surface forms to the corresponding entities.

Overlap Resolution.

At this point, many of the linked mentions will overlap each other. Dynamic programming resolves these conflicts by choosing the best-scoring set of non-overlapping mention-entity mappings. The experimental results show that resolving overlap improve the models performance consistently in different settings.

3. SYSTEM IMPLEMENTATION

Our database is constructed from both Wikipedia and Freebase. The whole system is implemented in C#.

Entity linking systems often require a large amount of memory due to the size of the structured/unstructured data for many entities. High memory consumption restricts the scale of an entity linking system, limiting the number of allowed entities that can be handled. Long loading times also

reduce the efficiency of conducting experiments. In E2E, we adopt the completion trie data structure proposed in [4] instead of a hash map dictionary. The completion trie greatly reduces the memory footprint and loading time of E2E.

We have tested two learning methods when developing E2E: a structured support vector machine algorithm [2] and a fast implementation of the MART gradient boosting algorithm [3]. The structural SVM model is a linear model that takes into account all of the candidates together in the same tweet. MART learns an ensemble of decision/regression trees with scalar values at the leaves, but treats each candidate separately. The submitted results are generated using MART due to its superior performance on our development set.

Features.

Three groups of features were used in our system. The textual features are the features regarding the textual properties of the surface form and its context. For example, one feature indicates if the current surface form and the surrounding words are capitalized or not. We also use features generated from the output of the in-house named entity recognition system that is specially designed to be robust on non-capitalized words. The entity graph features capture the semantic cohesiveness between the entity-entity and entity-mention pairs. This group of features was mainly calculated using the entity database and its structured data. Finally, the statistical features indicates the word usage and entity popularity using the information collected from the web.

Among the three group features, the statistical feature group is the most important one. We describe some of the most important features in the following. Let a denote the surface form of a candidate, and e denote the an entity. One important feature is the link probability feature $P_l(a)$, which indicates the probability that a phrase is used as an anchor in Wikipedia. For each phrase a, we also collect statistics about the probability that a phrase is capitalized in Wikipedia. We refer to this feature as the capitalization rate feature, $P_c(a)$.

We also compute features that captures the relationships between an anchor a and an entity e. The probability $P(e|a)$ captures the likelihood of an anchor linked to an Wikipedia entity. We have downloaded Wikipedia page view counts, representing page view information from 2012.[1] According to the popularity information, we add another probability feature that captures the relative popularity of the pages that could be linked from the anchor a. More precisely, $P_v(e|a) = v(e_i)/(\sum_{\{e \in \mathcal{E}(a) \cap \emptyset\}} v(e))$, where $v(e)$ represents the view count for the page e.

4. RESULTS

In our experiments, we split the training set into two sets that contains 1534 and 800 tweets, respectively. The 800-tweet data is used as our development set. Our analysis shows that robust mention detection is often the source of errors in the current the entity linking systems. In order to achieve better F1 score, we change the prediction function to

$$\arg\max_{e \in \{\mathcal{E}(a) \cap \emptyset\}} f(\Phi(m, a, e)) - s[e = \emptyset], \qquad (2)$$

[1] `http://dammit.lt/wikistats`

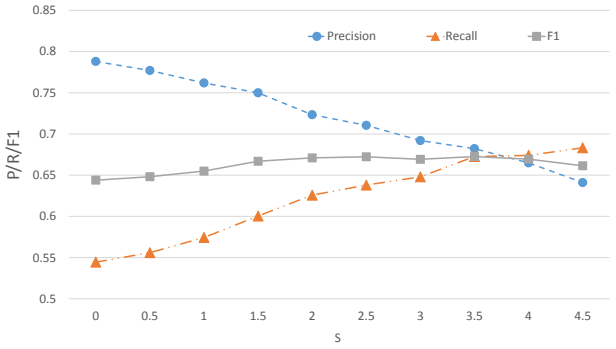


Figure 1: Results of E2E on the development set.

where $[\cdot]$ is an indicator function. When s increases, the system will produce more entities. From the results in Figure 1, we found that tuning s does impact results significantly. After learning parameters and desired value of s are chosen, we then retrain the E2E using the full training data, and generate final results with $s = 0$, 2.5 and 3.5, respectively.

Error Analysis.

We analyze at our results on the development set with $s = 3.5$. In the development set, there are 1304 mentions, and E2E generates total number of 18746 candidates in the candidate generation stage. Our error analysis shows that E2E misses 340 entity mentions and predict extra 284 mentions. Among the errors, E2E has troubles on the "number" entities (e.g. `1_(number)`). Further investigation reveals that the tokenization choice of E2E plays a big part of these errors, given that most punctuations are being treated as separate tokens. Interestingly, E2E only makes 44 cases where it correctly recognizes the mentions but link to wrong entities. Most errors occur when E2E fail to recognize mentions correctly.

5. CONCLUSIONS

In this paper, we presented E2E, a system that performs joint entity recognition and disambiguation on short and noisy text. We found that the substance of a successful entity linking system consists of successfully combining all of the components.

Due to the time limitation, the submitted system still has plenty of room to improve. For example, one important direction is to explore the relationships between different tweets to improve entity linking results. Developing a robust mention detection algorithm is an important research direction as well.

6. REFERENCES

[1] A. E. Cano Basave, G. Rizzo, A. Varga, M. Rowe, M. Stankovic, and A.-S. Dadzie. Making Sense of Microposts (#Microposts2014) Named Entity Extraction & Linking Challenge. In *Proc., #Microposts2014*, pages 54–60, 2014.

[2] M.-W. Chang and W.-T. Yih. Dual coordinate descent algorithms for efficient large margin structured prediction. *TACL*, 2013.

[3] J. H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 1999.

[4] B.-J. P. Hsu and G. Ottaviano. Space-efficient data structures for top-k completion. In *WWW*, 2013.

Named Entity Extraction and Linking Challenge: University of Twente at #Microposts2014

Mena B. Habib
Database Chair
University of Twente
Enschede, The Netherlands
m.b.habib@ewi.utwente.nl

Maurice van Keulen
Database Chair
University of Twente
Enschede, The Netherlands
m.vankeulen@utwente.nl

Zhemin Zhu
Database Chair
University of Twente
Enschede, The Netherlands
z.zhu@utwente.nl

ABSTRACT

Twitter is a potentially rich source of continuously and instantly updated information. Shortness and informality of tweets are challenges for Natural Language Processing (NLP) tasks. In this paper we present a hybrid approach for Named Entity Extraction (NEE) and Linking (NEL) for tweets. Although NEE and NEL are two topics that are well studied in literature, almost all approaches treated the two problems separately. We believe that disambiguation (linking) could help improving the extraction process. We call this potential for mutual improvement, the reinforcement effect. It mimics the way humans understand natural language. Furthermore, our proposed approaches handles uncertainties involved in the two processes by considering possible alternatives.

Categories and Subject Descriptors

H.3.1 [**Content Analysis and Indexing**]: Linguistic processing; I.7 [**Document and Text Processing**]: Miscellaneous

General Terms

Algorithms

Keywords

Named Entity Extraction, Named Entity Linking, Social Media Analysis, Twitter Messages.

1. INTRODUCTION

Named Entity Extraction (NEE) is a subtask of IE that aims to locate phrases (mentions) in the text that represent names of persons, organizations or locations regardless of their type. It differs from the term Named Entity Recognition (NER) which involves both extraction and classification into set of predefined classes. Named Entity Linking (NEL) (aka Named Entity Disambiguation) is the task of exploring which correct person, place, event, etc. is referred to by a mention. Wikipedia articles or Knowledge bases (KB) that is derived from Wikipedia are widely used as entities' references. NEE & NEL in tweets are challenging. The informal language of tweets plus their shortness make NEE & NEL processes more difficult.

Published as part of the #Microposts2014 Workshop proceedings, available online as CEUR Vol-1141 (http://ceur-ws.org/Vol-1141)

#Microposts2014, April 7th, 2014, Seoul, Korea.

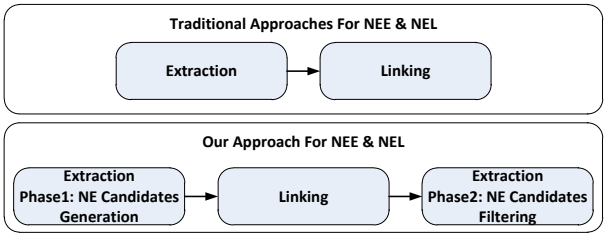


Figure 1: Traditional approaches versus our approach for NEE & NEL.

According to a literature survey, almost no research tackled the combined problem of NEE & NEL. Researchers either focus on NEE or NEL but not both. Systems that do NEL like AIDA [7], either require manual annotations for NE or use some off-the-shelf extraction models like Stanford NER [2]. Here, we present a combined approach for NEE and NEL for tweets with an application on #Microposts 2014 challenge [1]. Although the logical order for such system is to do extraction first then the disambiguation, we start with an extraction phase which aims to achieve high recall (find as much NE candidates as possible). Then we apply disambiguation for all the extracted mentions. Finally, we filter those extracted NE candidates into true positives and false positives using features derived from the disambiguation phase in addition to other word shape and KB features. The potential of this order is that the disambiguation step gives extra information about each NE candidate that may help in the decision whether or not this candidate is a true NE. Figure 1 shows our system architecture versus traditional one.

2. OUR APPROACH

2.1 NE Candidates Generation

For this task, we unionize the output of the following candidates generation methods:

- **Tweet Segmentation**: Tweet text is segmented using the segmentation algorithm described in [6]. Each segment is considered a NE candidate.
- **KB Lookup**: We scan all possible n-grams of the tweet against the mentions-entities table of DBpedia. N-grams that matches a DBpedia mention are considered NE candidates.
- **Regular Expressions**: We used regular expressions to extract numbers, dates and URLs from the tweet text.

2.2 NE Linking

Our NEL approach is composed of three steps; matcher, feature extractor, and SVM ranker.

- **Matcher**: This module takes each extracted mention candidate and looks for its Wikipedia reference candidates on DBpedia. Furthmore, for those mention candidates which don't have reference candidates in DBpedia, we use Google Search API to find possible Wikipedia pages for these mentions. This search helps to find references for misspelled or concatenated mentions like '*justinbieber*' and '*106andpark*'.
- **Feature Extractor**: This module is responsible for extracting a set of contextual and URL features for each candidate Wikipedia page as described in [3]. These features give indicators on how likely the candidate Wikipedia page could be a representative to the mention.
- **SVM Ranker**: After extracting the aforementioned set of features, SVM classifier is trained to rank candidate Wikipedia pages of a mention. For the challenge, we pick the page on the 1st order as a reference for the mention. The DBpedia URI is then generated from the selected Wikipedia URL.

2.3 NE Candidates Filtering

After generating the candidates list of NE, we apply our NE linking approach to disambiguate each extracted NE candidate. After the linking phase, we use SVM classifier to predict which candidates are true positives and which ones are not. We use the following set of features for each NE candidate to train the SVM:

- **Shape Features**: If the NE candidate is initially or fully capitalized and if it contains digits.
- **Probabilistic Features**:
  - The joint and the conditional probability of the candidate obtained from Microsoft Web N-Gram services.
  - The stickiness of the candidate as described in [6].
  - The candidate's frequency over around 5 million tweets[1].
- **KB Features**:
  - If the candidate appears in WordNet.
  - If the candidate appears as a mention in DBpedia KB.
- **Disambiguation Features**:
  - All the features used in the linking phase as described in [3]. We used only the feature set for the first top ranked entity page selected for the given NE candidate.

2.4 Final NE Set Generation

Beside the SVM, we also train a CRF model for NEE. We used the CRF model described in [4]. To generate the final NE set, we take the union of the CRF annotation set and SVM results, after removing duplicate extractions, to get the final set of annotations. We tried two methods to resolve overlapped mentions. In the first method (used in UTwente_Run1.tsv), we select the mention that appears in Yago KB [5]. If both mentions appear in Yago or both don't, we select the one with the longer length. In the second method (used in UTwente_Run2.tsv), we select only the mention with the longer length among the two overlapped mentions. The results shown in the next section are the results of the first method.

The idea behind this unionization is that SVM and CRF work in a different way. The former is a distance based classifier that uses numeric features for classification which CRF can not handle, while the latter is a probabilistic model that can naturally consider state-to-state dependencies and feature-to-state dependencies. On the other hand, SVM does not consider such dependencies. The hybrid approach of both makes use of the strength of each.

[1] `http://wis.ewi.tudelft.nl/umap2011/` + TREC 2011 Microblog track collection.

3. EXPERIMENTAL RESULTS

In this section we show our experimental results of the proposed approaches on the challenge training data [1] in contrast with other competitors. All our experiments are done through a 4-fold cross validation approach for training and testing. Table 1 shows the results of '**Our Linking Approach**' presented in section 2.2, in comparison with two modes of operation of AIDA [7]. The first mode is '**AIDA Cocktail**' which makes use of several ingredients: the prior probability of an entity being mentioned, the similarity between the context of the mention in the text and an entity, as well as the coherence among the entities. While the second mode is '**AIDA Prior**' which makes use only of the prior probability. The results show the percentage of finding the correct entity of the ground truth mentions. Table 2 shows the NEE results along the extraction process phases in contrast with '**Stanford NER**' [2]. Finally, table 3 shows our final results of both extraction and entity linking in comparison with our competitor ('**Stanford + AIDA**') where '**Stanford NER**' is used for NEE and '**AIDA Cocktail**' is used for NEL.

Table 1: Linking Results

	Percentage
Our Linking Approach	**70.98%**
AIDA Cocktail	56.16%
AIDA Prior	55.63%

Table 2: Extraction Results

	Pre.	**Rec.**	**F1**
Candidates Generation	0.120	**0.945**	0.214
Candidates Filtering (SVM)	**0.722**	0.544	0.621
CRF	0.660	0.568	0.611
Final Set Generation	0.709	0.706	**0.708**
Stanford NER	0.716	0.392	0.507

Table 3: Extraction and Linking Results

	Pre.	**Rec.**	**F1**
Extraction + Linking	**0.533**	**0.534**	**0.534**
Stanford + AIDA	0.509	0.279	0.360

4. REFERENCES

[1] A. E. Cano Basave, G. Rizzo, A. Varga, M. Rowe, M. Stankovic, and A.-S. Dadzie. Making Sense of Microposts (#Microposts2014) Named Entity Extraction & Linking Challenge. In *Proc., #Microposts2014*, pages 54–60, 2014.

[2] J. R. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proc. of ACL 2005*, pages 363–370, 2005.

[3] M. B. Habib and M. van Keulen. A generic open world named entity disambiguation approach for tweets. In *Proc. of KDIR 2013*, pages 267–276, 2013.

[4] M. B. Habib, M. van Keulen, and Z. Zhu. Concept extraction challenge: University of Twente at #MSM2013. In *Proc., #MSM2013*, pages 17–20, 2013.

[5] J. Hoffart, F. M. Suchanek, K. Berberich, E. L. Kelham, G. de Melo, and G. Weikum. Yago2: Exploring and querying world knowledge in time, space, context, and many languages. In *Proc. of WWW 2011*, 2011.

[6] C. Li, J. Weng, Q. He, Y. Yao, A. Datta, A. Sun, and B.-S. Lee. Twiner: named entity recognition in targeted twitter stream. In *Proc. of SIGIR 2012*, pages 721–730, 2012.

[7] M. A. Yosef, J. Hoffart, I. Bordino, M. Spaniol, and G. Weikum. Aida: An online tool for accurate disambiguation of named entities in text and tables. *PVLDB*, 4(12):1450–1453, 2011.

DataTXT at #Microposts2014 Challenge

Ugo Scaiella
Michele Barbera
Stefano Parmesan
Spaziodati Srl
Trento, Italy
{surname}@spaziodati.eu

Gaetano Prestia
Net7 Srl
Pisa, Italy
prestia@netseven.it

Emilio Del Tessandoro
Mario Verì
Dipartimento di Informatica
University of Pisa, Italy
deltessa@di.unipi.it
veri@di.unipi.it

ABSTRACT

In this paper we describe the approach taken for the "Making Sense of Microposts challenge 2014" (#Microposts2014), where participants were asked to cross reference micro-posts extracted from Twitter with DBpedia URIs belonging to a given taxonomy.

For this task we deployed DATATXT[1] which is the evolution of TAGME[3], the state-of-the-art topic annotator for short texts and which has proven to be very effective and efficient in several challenging scenarios[2].

Keywords

topic annotator, entity extraction, datatxt

1. INTRODUCTION

The #Microposts2014 challenge[1] focuses on the task of annotating micro-posts with DBpedia entities belonging to a given taxonomy. With respect to traditional Information Retrieval tasks, such data poses new challenges in terms of the effectiveness and efficiency of the algorithms and applications because data is so short and noisy that it is difficult to mine significant statistics that are rather available when texts are long and well written. Additionally, participants have to deal with the issue of associating extracted entities with the provided taxonomy, that makes this challenge even harder.

For this challenge, we deployed DATATXT, an entity extraction system that is the evolution of TAGME[3]. An instance of DATATXT has been specifically trained using the official training set provided in this challenge.

2. ANATOMY OF DATATXT

DATATXT is able to identify meaningful sequences of one or more terms in unstructured texts on-the-fly and with high accuracy, and link them to a pertinent Wikipedia page. DATATXT maintains the core algorithm of its predecessor, TAGME, but adds functionality and several improvements in terms of cleaning the input text and identifying mentions.

The algorithm is based on the anchor texts drawn from Wikipedia for identifying mentions in input text. When an input text is received, it judiciously cross-references each anchor a found in the input text T with one pertinent page p_a of Wikipedia. DATATXT first identifies for each anchor a all possible pages p_a linked by a in Wikipedia. Then, from these pages, it selects the best association $a \mapsto p_a$ by computing a score based on a "collective agreement" between the page p_a and the pages p_b that can be associated with all other anchors $b^1 ... b^n$ detected in T. We deploy a voting-schema, where pages p_b vote for each candidate p_a according to a function that estimates the relatedness between two Wikipedia pages by exploiting the underlying graph. Further details of this voting-schema and the relatedness function can be found in [3]. Not all mentions extracted in this way are worth annotating, so a confidence score is assigned to all mentions. This score is based on (a) a–priori statistics based on Wikipedia and (b) other figures representing the coherence of the candidate entity with respect to the whole text. It is thus possible to discard those whose confidence score is below a given threshold.

DATATXT does not rely on any linguistic feature, but only on statistics and data extracted from Wikipedia. We argue that this approach, derived from TAGME, yields better results when dealing with user generated content such as micro-posts, where well-known NLP tools, such as part–of–speech taggers, are less effective because texts are short, fragmented and often contain slang and/or misspelled words. An in-depth evaluation of TAGME's effectiveness and comparison with others annotators was recently published in [2], showing the validity of this approach.

3. TRAINING

DATATXT was designed for short texts, but it is effective for long texts as well[2]. However there are some parameters that can be amended in order to better fit the context of this challenge. One of them is ϵ, which is used to tune the disambiguation algorithm[3] and defines whether DATATXT should rely more on the context or favor more common topics in order to discover entities. Using a higher value favors more common topics, which may lead to better results when processing fragmented inputs where the context is not always reliable. Two other parameters have been taken into account: (a) the minimum link probability, say δ, that is used to discard a mention that is rarely used as

[1] http://dandelion.eu/datatxt

Published as part of the #Microposts2014 Workshop proceedings, available online as CEUR Vol-1141 (http://ceur-ws.org/Vol-1141)

#Microposts2014, April 7th, 2014, Seoul, Korea.

anchor texts in Wikipedia; (b) the minimum commonness, say γ, that is used to discard a possible association $a \mapsto p_a$, thus reducing the "ambiguity" of a mention. Refer to [3], for further details on these two thresholds. DATATXT assigns a confidence score to each annotation so that those that are below a given threshold, say ϕ, can be discarded. This parameters can be used to balance precision vs. recall and the best value may vary based on the application context. For each configuration we tested, we evaluated the results using 20 values of this threshold, ranging from 0 to 1.

Another important issue we faced, is that the annotation task of this challenge has been restricted to entities belonging to a limited taxonomy. DATATXT is a generic *topic* annotator and it extracts all topics contained in the input text. If we considered the overall output produced by DATATXT, the results, and in particular the precision, would be significantly penalized because DATATXT also includes *topics* that are not part of the taxonomy. As an example consider this tweet, which was part of the training set: "*Bank of America posts $8.8 billion loss in second quarter due to mortgage security settlement*". The human annotators extracted *Bank of America* as the only mention of this micropost, whereas DATATXT extracts also *mortgage* and *security* linking them to `Mortgage_loan` and `Security_(finance)` respectively. These are not errors of the system, but #MSM2014 focused on a limited taxonomy and *mortgage* and *security* are not part of it. Unfortunately, given a DBpedia URI it was not possible to automatically check whether or not the entity belongs to that taxonomy. To address the issue, we initially tested a naive approach using a white–list of entities derived from the training set. This is useful but, of course, is not generic. Thus, we designed another approach that provides the probability that a generic entity belongs to the taxonomy based on the Wikipedia categories and DBpedia types associated with the entity. We thus gathered all Wikipedia categories and all DBpedia types associated with each entity extracted by DATATXT from the training set. We then counted the occurrences of categories and types for all entities that were part of the ground-truth and the occurrences of categories and types for those that were not. For each category/type we then computed a probability. Given an entity e extracted by DATATXT, we thus computed the probability that e belongs to the taxonomy by computing a weighted sum of probabilities of all categories and types of e. Finally, we discarded from the results all entities whose probability is below a given threshold. The value of this threshold, called β, was experimentally evaluated using the training set, together with other parameters mentioned above. We also tested a third approach by deploying a C4.5 classifier, which was trained by exploiting types and categories derived as mentioned above. Categories and types were thus deployed as features to train the classifier.

For parameters tuning, we simply used a grid search in an N–dimensional space, using a 5-fold cross evaluation, in order to avoid over–fitting. Note that DATATXT is very efficient, and the evaluation of a single parameter combination (ie the annotation of more than 2K tweets) takes about 800 ms, thus this search is feasible even with a long list of combinations. Tuned values of parameters do not change significantly across the different folds, showing a good stability and generality of the approach (see Table 1).

Fold#	β	γ	δ	ϵ	ϕ	F_1
1	0.4	0.4	0.3	0.15	0.7	0.5985
2	0.4	0.4	0.2	0.2	0.65	0.5853
3	0.3	0.5	0.2	0.2	0.7	0.5722
4	0.4	0.5	0.2	0.2	0.7	0.5690
5	0.4	0.4	0.2	0.2	0.6	0.5737

Table 1: Tuning second approach, results per single fold of cross evaluation

Approach	Precision	Recall	F_1
1. White–list only	66.3	41.3	50.2
2. White–list + types prob.	65.6	50.7	57.2
3. White–list + C4.5 classifier	75.8	55.5	64.1

Table 2: Results of our approaches.

4. RESULTS

During the training phase, we noticed several differences between the annotation generated by DATATXT and the annotation provided in the ground-truth, therefore we implemented a few post–annotation steps to improve the performance for this challenge: (a) DATATXT does not annotate dates or numbers, so a step that identifies these types of mentions using simple regular expressions was added; (b) DATATXT annotates only the first occurrence of a mention, so a post–processing step to handle repeated mentions was added. These steps do not affect the core algorithm and thus were not considered during the training phase, however they improve the performance of DATATXT for this challenge. Table 2 shows the overall results of the cross evaluation of our approaches using the training set. These figures are not directly comparable those presented in [2], as #MSM2014 focused on a limited set of entities, i.e. the ones specified by the taxonomy.

5. CONCLUSIONS

We have described the approach taken by our group for the #MSM2014 challenge, where we deployed DATATXT, the evolution of the state-of-the-art topic annotator TAGME. Given that its algorithm does not depend on linguistic features, DATATXT is very accurate even in this scenario. We have also outlined a basic approach to verticalize the general–purpose extraction algorithm to improve the performance in the domain defined within this challenge. We believe that this approach to verticalization could be further refined by applying more sophisticated machine–learning techniques, such as SVM or CRF.

6. REFERENCES

[1] A. E. Cano Basave, G. Rizzo, A. Varga, M. Rowe, M. Stankovic, and A.-S. Dadzie. Making Sense of Microposts (#Microposts2014) Named Entity Extraction & Linking Challenge. In *Proc., #Microposts2014*, pages 54–60, 2014.

[2] M. Cornolti, P. Ferragina and M. Ciaramita. A framework for benchmarking entity-annotation systems. In *WWW*, 249–260, 2013.

[3] P. Ferragina and U. Scaiella. Fast and Accurate Annotation of Short Texts with Wikipedia Pages. *IEEE Software*, 29(1): 70–75, 2012.

Adapting AIDA for Tweets

Mohamed Amir Yosef, Johannes Hoffart, Yusra Ibrahim,
Artem Boldyrev, Gerhard Weikum
Max Planck Institute for Informatics
Saarbrücken, Germany
{mamir|jhoffart|yibrahim|boldyrev|weikum}@mpi-inf.mpg.de

ABSTRACT

This paper presents our system for the "Making Sense of Microposts 2014 (#Microposts2014)" challenge. Our system is based on AIDA, an existing system that links entity mentions in natural language text to their corresponding canonical entities in a knowledge base (KB). AIDA collectively exploits the prominence of entities, contextual similarities, and coherence to effectively disambiguate entity mentions. The system was originally developed for clean and well-structured text (e. g. news articles). We adapt it for microposts, specifically tweets, with special focus on the named entity recognition and the entity candidate lookup.

Keywords

Entity Recognition, Entity Disambiguation, Social Media

1. INTRODUCTION

Microblogs present a rich field for harvesting knowledge, especially Twitter with more than 500 million tweets per day [5]. However, extracting information from short informal microposts (tweets) is a difficult task due to insufficient contextual evidence, typos, cryptic abbreviations, and grammatical errors. The MSM challenge adresses a fundamental task for knowledge harvesting, namely Named Entity Recognition and Disambiguation (NERD). The goal is to identify entity mentions in text and link them to canonical entities in (mostly Wikipedia-derived) KBs such as `www.yago-knowledge.org` or `dbpedia.org`. We participate in the MSM challenge with an adaptation of the existing AIDA [4] system, a robust NERD framework originally designed for handling input texts with clean language and structure, such as news articles. We adapt it to handle short microposts by adding additional components for named entity recognition, name normalization, and extended candidate entity retrieval. We also integrate data harvested from Twitter API into our model to cope with the context sparsity. Moreover, we tuned the AIDA algorithm parameters to accommodate the brief informal nature of tweets. In the following sections we will first briefly introduce AIDA, then present our approach for adapting AIDA to microblogs, and finally detail our experimental settings.

Published as part of the #Microposts2014 Workshop proceedings, available online as CEUR Vol-1141 (`http://ceur-ws.org/Vol-1141`)

#Microposts2014, April 7th, 2014, Seoul, Korea.

2. AIDA FRAMEWORK OVERVIEW

The AIDA framework deals with arbitrary text that contains mentions of named entities (people, music bands, universities, etc.), which are detected using the Stanford Named Entity Recognition (NER) [2]. Once the names are detected, the entity candidates are retrieve by a dictionary lookup, where the dictionary is compile from Wikipedia redirects, disambiguation pages, and link anchors. For the actual disambiguation, we construct a weighted mention-entity graph containing all mentions and candidates present in the input texts as nodes. The graph contains two kinds of edges: **mention-entity edges:** between mentions and their candidate entities, weighted with the *similarity* between a mention and a candidate entity, and **entity-entity edges:** between different entities with weights that capture the *coherence* between two entities.

The actual disambiguation in form of mention-entity pairs is obtained by reducing this graph into a dense sub-graph where each mention is connected to exactly one candidate entity. The *similarity* between a mention and a candidate entity is computed as a linear combination of two ingredients: 1) the prior probability of a entity given a mention, which is estimated from the fraction of a Wikipedia link anchor (the mention) pointing to a given article (the entity); 2) based on the partial overlap between mention's context (the surrounding text) and a candidate entity's context (a set of keyphrases gathered from Wikipedia). For entity-entity edges we harness the Wikipedia link structure to estimate *coherence* weights. We define the coherence between two entities to be proportional to the number of Wikipedia articles at which they were co-referenced [6]. More details on the features, algorithms and implementation of this approach are included in [4, 7].

3. ADAPTING AIDA FOR TWEETS

AIDA was geared for well-written and long texts, such as news articles. We made the following modifications to adapt it for tweets.

Named Entity Recognition. AIDA originally uses Stanford NER, with a model trained on newswire snippets, a perfect fit for news texts. However, it is not optimized for handling user generated content with typos and abbreviations. Hence, we employ two different components for

mention detection: The first is Stanford NER with models trained for caseless mention detection; the second is our in-house dictionary-based NER tool. The dictionary-based NER is performed in two stages:

1. Detection of named entity candidates using dictionaries of all names of all entities in our knowledge base, using partial prefix-matches for lookups to allow for shortening of names or little differences in the later part of a name. For example, we would recognize the ticker symbol "GOOG" even though our dictionary only contains "Google". The character-wise matching of all names of entities in our KB is efficiently implemented using a prefix-tree data structure.
2. The large number of false positives are filtered using a collection of heuristics, e. g. the phrase has to contain a NNP tag or it has to end with a suffix signifying a name such as "Ave" in "Fifth Ave".

Mention Normalization. The original AIDA did not distinguish between the textual representation of the mention, and its normalized form that should be used to query the dictionary. For example, the hashtag `"#BarackObama"` should be normalized to "Barack Obama" before matching it against the dictionary. Furthermore, many mentions of named entities are referred to in the tweet by their Twitter user ID, such as `"@EmWatson"` the Twitter account of the British actress "Emma Watson". Because the Twitter user IDs are not always informative we access the account metadata, which contains the full user name most of the time. In fact, we attach to each mention string a set of normalized mentions and use all of them to query the dictionary. For example `"@EmWatson"` will have the following normalized mentions {"EmWatson", "Em Watson", "Emma Watson"}, and each of them will be matched against the dictionary to retrieve the set of candidate entities. As the prior probability is on a per-mention basis, we compute the aggregate prior probability of an entity e_i given a mention m_i:

$$prior(m_i, e_i) = \max_{m' \in N(m_i)} prior(m', e_i) \quad (1)$$

where $N(m_i)$ is the set of normalized mentions of m_i. The maximum is taken in order not to penalize an entity if one of the normalized mentions is rarely used to refer to it.

Approximate Matching. This step is employed iff the previous normalization step did not produce candidate entities for a given mention. For example, it is not trivial to automatically split a hashtag like `"#londonriots"`, and hence its normalized mention set, {"londonriots"}, does not have any candidate entities. We address this by representing both the mention strings and dictionary keys as vectors of character-trigrams between which the cosine similarity is computed. We only consider the candidate entity if cosine similarity between the mention and candidate entity keys is above a certain threshold (experimentally determined as 0.6).

Parameter Settings. In our graph representation, the weight of a mention-entity edge is computed by a linear combination of different similarity measures. To estimate the constants of the linear combination, we split the provided tweets training dataset into `TRAIN` and `DEVELOP` chunks, using `TRAIN` for the estimation. We estimated further hyper-parameters for our algorithm (like the importance of mention-entity vs. entity-entity edges) on `DEVELOP`.

Unlinkable Mentions. Some mentions should not be disambiguated to a entity, even though there are candidates for it. This is especially frequent in the case of social media, where a large number of user names are ambiguous but do not refer to any existing KB entity – imagine how many Will Smiths exists besides the famous actor. We address this problem by thresholding on the disambiguation confidence as defined in [3], where a mention is considered unlinkable and thus removed if the confidence is below a certain threshold, estimated as 0.4 on `DEVELOP`.

4. EXPERIMENTS

We conducted our experiments on the dataset provided in [1]. We carried out experiments with three different setups. First we used Stanford NER trained for entity detection, along with mention prior probability and key-phrases matching for entity disambiguation. In the second experiment we added coherence graph disambiguation to the previous setting. The third setting is similar to the first one, but we use our dictionary-based NER instead of Stanford's for entity detection. Note that we automatically annotate all digit-only tokens as mentions using a regular expression, as all numbers were annotated in the training data. The results of running the three experiments on the testing dataset are correspondingly provided with the following ids: AIDA_1, AIDA_2 and AIDA_3.

During our experiments, our runs achieved around 51% F1 on the `DEVELOP` part of the training data, where a mention is counted as true positive only if both the mention span matches the ground truth perfectly and the entity label is correct.

5. CONCLUSION

AIDA is a robust framework that can be adapted to any type of natural language text, here we use it to disambiguate names to entities in tweets. We found that using a dictionary-based NER worked well for the sometimes ill-formatted inputs. An approximate candidate lookup crucially improves recall, which in combination with discarding low-confidence mentions improves the results.

6. REFERENCES

[1] A. E. Cano Basave, G. Rizzo, A. Varga, M. Rowe, M. Stankovic, and A.-S. Dadzie. Making Sense of Microposts (#Microposts2014) Named Entity Extraction & Linking Challenge. In #Microposts2014.

[2] J. R. Finkel, T. Grenager, C. Manning. Incorporating Non-local Information into Information Extraction systems by gibbs sampling. ACL 2005

[3] J. Hoffart, Y. Altun, G. Weikum. Discovering Emerging Entities with Ambiguous Names. WWW 2014

[4] J. Hoffart, M. A. Yosef, I. Bordino et al. Robust Disambiguation of Named Entities in Text. EMNLP 2011

[5] R. Holt. Twitter in numbers, March 2013.

[6] D. Milne, I. Witten. An Effective, Low-Cost Measure of Semantic Relatedness Obtained from Wikipedia Links. WIKIAI workshop at AAAI 2008

[7] M. A. Yosef et al. AIDA: An online tool for accurate disambiguation of named entities in text and tables. VLDB 2011

Section IVb:

NEEL Challenge Submissions II Posters

Linking Entities in #Microposts

Romil Bansal, Sandeep Panem, Priya Radhakrishnan,
Manish Gupta, Vasudeva Varma
International Institute of Information Technology, Hyderabad

ABSTRACT

Social media has emerged to be an important source of information. Entity linking in social media provides an effective way to extract useful information from microposts shared by the users. Entity linking in microposts is a difficult task as they lack sufficient context to disambiguate the entity mentions. In this paper, we do entity linking by first identifying entity mentions and then disambiguating the mentions based on three different features: (1) similarity between the mention and the corresponding Wikipedia entity pages; (2) similarity between the mention and the tweet text with the anchor text strings across multiple webpages, and (3) popularity of the entity on Twitter at the time of disambiguation. The system is tested on the manually annotated dataset provided by Named Entity Extraction and Linking (NEEL) Challenge 2014, and the obtained results are on par with the state-of-the-art methods.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms, Experimentation

Keywords

Named Entity Extraction and Linking (NEEL) Challenge, Entity Linking, Entity Disambiguation, Social Media

1. INTRODUCTION

Social media networks like Twitter have emerged to be major platforms for sharing information in form of short messages (tweets). Analysis of tweets can be useful for various applications like e-commerce, entertainment, recommendations, etc. Entity linking is the one such analysis task which deals with finding correct referent entities in the knowledge base for various mentions in the tweet. Entity linking in social media is important as it helps in detecting, understanding and tracking information about an entity shared across social media.

Published as part of the #Microposts2014 Workshop proceedings, available online as CEUR Vol-1141 (http://ceur-ws.org/Vol-1141)

#Microposts2014, April 7th, 2014, Seoul, Korea.

Entity linking consists of two different tasks, mention detection and entity disambiguation. Entity linking from general text is a well explored problem. Existing entity linking tools are intended for use over news corpora and similar document-based corpora with relatively long length. But as microposts lack sufficient context, these context-based approaches fail to perform well on microposts.

In this paper we describe our system proposed for the NEEL Challenge 2014 [1]. The proposed system disambiguates the entity mentions in the tweets based on three different measures: (1) Wikipedia's context based measure (§2.2.1); (2) anchor text based measure (§2.2.2); and (3) Twitter popularity based measure (§2.2.3).

The mention detection is done using existing Twitter part-of-speech (POS) taggers [2, 5].

2. OUR APPROACH

2.1 Mention Detection

Mention detection is the task of finding entity mentions in the given text. We assumed mentions as named entities present inside the tweets. Various approaches for named entity recognition in tweets have been proposed recently [3, 5]. This includes spotting continuous sequence of proper nouns as named entities in the tweet. But sometimes named entities like 'Statue of Liberty', 'Game of Thrones' etc. also includes tokens other than nouns. To detect such mentions, Ritter *et al.* [5] proposed a machine learning based system for named entity detection in tweets. Gimpel *et al.* [2] present yet another approach for POS tagging of tweets. We tried both of these POS taggers to extract proper noun sequences. In our experiments Ritter *et al.*'s tagger gave an accuracy of 77% while Gimpel *et al.*'s tagger gave an accuracy of 92%. So we merged the results from both as shown in Fig. 1. The tweet text is fed to the system and the longest continuous sequences of proper noun tokens detected using the above approach are extracted as the entity mentions from the given tweet. The merged system provided an accuracy of 98% in predicting mentions.

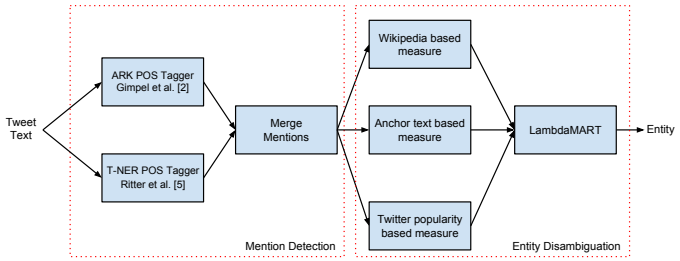


Figure 1: System Architecture

2.2 Entity Disambiguation

Entity disambiguation is the task of assigning the correct referent entity from the knowledge base to the given mention. We disambiguate the entity mention using three measures as described below. The scores from these three measures are combined using LambdaMART [7] model to arrive at the final disambiguated entity.

2.2.1 Wikipedia's Context based Measure (M1)

This measure disambiguates a mention by calculating the frequency of occurrence of the mention in the Wikipedia corpus. Wikipedia's context based measure has been used in various approaches for disambiguating mentions in tweets [4]. We query MediaWiki API[1] with the entity mention. MediaWiki API returns the candidate entities in the ranked order. Each candidate entity is assigned its reciprocal rank as score. Thus, a ranked list of candidate entities with their scores are created using M1.

2.2.2 Anchor Text based Measure (M2)

Google Cross-Wiki Dictionary (GCD) [6] is a string to concept mapping, created using anchor text from various web pages. A concept is an individual Wikipedia article, identified by its URL. The text strings constitute the anchor hypertexts that refer to these concepts. Thus, anchor text strings represent a concept. We query the GCD with a mention along with the tweet text. Based on the similarity to the query string, a ranked list of probable candidate entities are created (which is the ranked list using M2). The ranking criteria is based on Jaccard similarity between the anchor text and the query. So if the mention is highly similar to the anchor text, then the corresponding concept will have a high score.

2.2.3 Twitter Popularity based Measure (M3)

Tweets about entities follow a bursty pattern. Bursty patterns are the bursts of tweets that appear after an event relating to an entity happens. We exploited this fact and tried to measure the number of times the given mention refers to a particular entity on Twitter recently. The mention is queried on Twitter API[2] and the resultant tweets are analyzed. All the tweets along with the mention are then queried on the GCD and the candidate entities are taken. Based on the scores returned using GCD, all the candidate entities are ranked (which is the ranked list using M3). As Twitter popularity based measure captures the people's interests at a particular time, it works well for entity disambiguation on recent tweets. In essence, the methods M2 and M3 are similar but with different inputs. Both use GCD, and produce candidate mentions and score as output. However, M2 takes mention and single tweet text as input whereas M3 takes mention and multiple tweets as input.

We have three rankings available using M1, M2, M3. Now the task is to arrive at the final ranking of the candidate entities by combining the rankings of the three different models. The rankings of different models should be combined such that the overall F1 score is maximized. For this, we use LambdaMART which combines LambdaRank and MART models. LambdaMART creates boosted regression trees for combining the rankings of the three different systems.

3. RESULTS AND EVALUATION

The dataset comprises of 2.3K tweets each annotated with the entity mention and its corresponding DBpedia URL. We divided the dataset into the 7:3 (train:test) ratio. Table 1 shows the results obtained using the NEEL Challenge evaluation framework. The best results are obtained when a combination of all the measures were used for disambiguation[3]. A 5-fold cross validation on the dataset gave an average **F1** of **0.52** for **M1+M2+M3**.

Table 1: Results: M1 represents Wikipedia's Context based Measure (§2.2.1), M2 represents Anchor Text based Measure (§2.2.2) and M3 represents Twitter Popularity based Measure (§2.2.3)

Measure	F1-measure
M1	0.355
M2	0.100
M3	0.194
M1+M2	0.355
M2+M3	0.244
M1+M3	0.405
M1+M2+M	**0.512**

4. CONCLUSION

For effective entity linking, mention detection in tweets is important. We improve the accuracy of detecting mentions by combining various Twitter POS taggers. We resolve multiple mentions, abbreviations and spell variations of a named entity using the Google Cross-Wiki Dictionary. We also use popularity of an entity on Twitter for improving the disambiguation. Our system performed well with a F1 score of 0.512 on the given dataset.

5. REFERENCES

[1] A. E. Cano Basave, G. Rizzo, A. Varga, M. Rowe, M. Stankovic, and A.-S. Dadzie. Making Sense of Microposts (#Microposts2014) Named Entity Extraction & Linking Challenge. In *Proc., 4th Workshop on Making Sense of Microposts (#Microposts2014)*, pages 54–60, 2014.

[2] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. A. Smith. Part-of-speech Tagging for Twitter: Annotation, Features, and Experiments. In *Proc. of the* 49^{th} *Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2 (NAACL-HLT)*, pages 42–47, 2011.

[3] S. Guo, M.-W. Chang, and E. Kıcıman. To Link or Not to Link? A Study on End-to-End Tweet Entity Linking. In *Proc. of the Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 1020–1030, 2013.

[4] X. Liu, Y. Li, H. Wu, M. Zhou, F. Wei, and Y. Lu. Entity Linking for Tweets. In *Proc. of the* 51^{th} *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1304–1311, 2013.

[5] A. Ritter, S. Clark, Mausam, and O. Etzioni. Named Entity Recognition in Tweets: An Experimental Study. In *Proc. of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2011.

[6] V. I. Spitkovsky and A. X. Chang. A Cross-Lingual Dictionary for English Wikipedia Concepts. In *Proc. of the* 8^{th} *Intl. Conf. on Language Resources and Evaluation (LREC)*, 2012.

[7] Q. Wu, C. J. Burges, K. M. Svore, and J. Gao. Adapting Boosting for Information Retrieval Measures. *Journal of Information Retrieval*, 13(3):254–270, Jun 2010.

[1] https://www.mediawiki.org/wiki/API:Search

[2] https://dev.twitter.com/docs/api/1.1/get/search/tweets

[3] submitted as Agglutweet_1.tsv

Part-of-Speech is (almost) enough: SAP Research & Innovation at the #Microposts2014 NEEL Challenge

Daniel Dahlmeier
SAP Research and Innovation
#14 CREATE, 1 Create Way
Singapore
d.dahlmeier@sap.com

Naveen Nandan
SAP Research and Innovation
#14 CREATE, 1 Create Way
Singapore
naveen.nandan@sap.com

Wang Ting
SAP Research and Innovation
#14 CREATE, 1 Create Way
Singapore
dean.wang@sap.com

ABSTRACT

This paper describes the submission of the SAP Research & Innovation team at the #Microposts2014 NEEL Challenge. We use a two-stage approach for named entity extraction and linking, based on conditional random fields and an ensemble of search APIs and rules, respectively. A surprising result of our work is that part-of-speech tags alone are almost sufficient for entity extraction. Our results for the combined extraction and linking task on a development and test split of the training set are 34.6% and 37.2% F_1 score, respectively, and for the test set is 37%.

Keywords

Conditional Random Field, Entity Extraction, DBpedia Linking

1. INTRODUCTION

The rise of social media platforms and microblogging services has led to an explosion in the amount of informal, user-generated content on the web. The task of the #Microposts2014 workshop NEEL challenge is named entity extraction and linking (NEEL) for microblogging texts [1]. Named-entity extraction and linking is a challenging problem because tweets can contain almost any content, from serious news, to personal opinions, to sheer gibberish and both extraction and linking have to deal with the inherent ambiguity of natural language.

In this paper, we describe the submission of the SAP Research & Innovation team. Our system breaks the task into two separate steps for extraction and linking. We use a conditional random field (CRF) model for entity extraction and an ensemble of search APIs and rules for entity linking. We describe our method and present experimental results based on the released training data. One surprising finding of our experiments is that part-of-speech tags alone perform almost as well as the best feature combinations for entity extraction.

Published as part of the #Microposts2014 Workshop proceedings, available online as CEUR Vol-1141 (http://ceur-ws.org/Vol-1141)

#Microposts2014, April 7th, 2014, Seoul, Korea.

2. METHOD

2.1 Extraction

We use a sequence tagging approach for entity extraction. In particular, we use a conditional random field (CRF) which is a discriminative, probabilistic model for sequence data with state-of-the-art performance [3]. A linear-chain CRF tries to estimate the conditional probability of a label sequence $\mathbf{y}$ given the observed features $\mathbf{x}$, where each label y_t is conditioned on the previous label y_{t-1}. In our case, we use BIO CoNLL-style tags [5]. We do not differentiate between different entity classes for BIO tags (e.g, 'B' instead of 'B-PERSON').

The choice of appropriate features can have a significant impact on the model's performance. We have investigated a set of features that are commonly used for named entity extraction. Table 1 lists the features. The casing features

Feature	Example
words	Obamah
words lower	obamah
POS	^
title case	True
upper case	False
stripped words	obamah
is number	False
word cluster	-NONE-
dbpedia	dbpedia.org/resource/Barack_Obama

Table 1: Examples of features for entity extraction.

upper case and *lower case* and the *is number* feature are implemented using simple regular expressions. The *stripped words* feature is the lowercased word with initial hashtags and @ characters removed. The DBpedia feature is annotated automatically using the DBpedia Spotlight web API [1] and acts as a type of gazetteer feature. For a label y_t at position t, we consider features x extracted at the current position t and previous position $t-1$. We experimented with larger feature contexts but they did not improve the result on the development set.

2.2 Linking

For the linking step, we explore different search APIs, such as Wikipedia search[2], DBpedia Spotlight, and Google search to retrieve the DBpedia resource for a mention. We begin with using the extracted entities individually as query terms

[1] github.com/dbpedia-spotlight/dbpedia-spotlight
[2] github.com/goldsmith/Wikipedia

Feature	F_1 score
POS	0.622
+ is number	**0.629**
+ upper case	0.623

Table 2: Results for extraction feature selection.

to these search APIs. As ambiguity is a major concern for the linking task, for tweets where there are multiple entities extracted, we use the entities combined as an additional query term. For example, a tweet with annotated entities as *Sean Hoare* and *phone hacking*, *Sean Hoare* would map to a specific resource in DBpedia but *phone hacking* could refer to more than one resource. By using the query term *"phone hacking + Sean Hoare"*, we can help boost the rank for the resource *"News International phone hacking scandal"* to map to the entity *phone hacking* instead of a general article on *"Phone Hacking"*. In our system, we make use of the Web APIs for Wikipedia search and DBpedia Spotlight together with some hand-written rules to rank the resources returned. The result of the ranking step is then used to construct the DBpedia resource URL to which the entity is mapped.

3. EXPERIMENTS AND RESULTS

In this section, we present experimental results of our method, based on the on the data released by the organizers.

3.1 Data sets

We split the provided data set into a training (first 60%), development (dev, next 20%), and test (dev-test, last 20%) set. We perform standard pre-processing steps. We perform tokenization and POS tagging using the Tweet NLP toolkit [4], lookup word cluster indicators for each token from the Brown clusters released by Turian *et al.* [6], and annotate the tweets with the DBpedia Spotlight web API.

3.2 Extraction

We train the CRF model on the training set of the data, perform feature selection based on the dev set, and test the resulting model on the dev-test set. We evaluate the resulting models using precision, recall, and F_1 score. In all experiments, we use the CRF++ implementation of conditional random fields[3] with default parameters. We found in initial experiments that the CRF parameters did not have a great effect on the final score. We employ a greedy feature selection method [2] to find the subset of the best features. Table 2 shows the results of the feature selection experiments on the development set. We can see that POS tags alone give a F_1 score of 62.2%. Adding the binary *is number* feature increases the score to 62.9%. Additional features, such as lexical features, word clusters, or the DBpedia Spotlight annotations, do not help and even decrease the score. Surprisingly the word token itself is *not* selected as one of the features. Thus, the CRF performs its task without even looking at the word itself! After feature selection, we retrain the CRF with the best performing feature set {*POS, is number*} and evaluate the model on the dev and dev-test set. The results are shown in Table 3.

[3] `code.google.com/p/crfpp/`

Data set	Precision	Recall	F_1 score
Dev	0.673	0.591	0.629
Dev-test	0.671	0.579	0.622

Table 3: Results for entity extraction.

3.3 Linking

To test our linking system, we follow two approaches. First, we measure the accuracy of the linking system using the gold standard where we observe an accuracy of 67.6%. As a second step, we combine the linking step with our entity extraction step and measure the F_1 score. Table 4 shows the results on the dev and dev-test split for the combined system.

Data set	Precision	Recall	F_1 score
Dev	0.436	0.287	0.346
Dev-test	0.477	0.304	0.372

Table 4: Results for entity extraction and linking.

4. CONCLUSION

We have described the submission of the SAP Research & Innovation team to the #Microposts2014 NEEL shared task. Our system is based on a CRF sequence tagging model for entity extraction and an ensemble of search APIs and rules for entity linking. Our experiments show that POS tags are a surprisingly effective feature for entity extraction in tweets.

5. ACKNOWLEDGEMENT

The research is partially funded by the Economic Development Board and the National Research Foundation of Singapore.

6. REFERENCES

[1] A. E. Cano Basave, G. Rizzo, A. Varga, M. Rowe, M. Stankovic, and A.-S. Dadzie. Making Sense of Microposts (#Microposts2014) Named Entity Extraction & Linking Challenge. In *Proc., #Microposts2014*, pages 54–60, 2014.

[2] A.L. Berger, V.J. Della Pietra, and S.A. Della Pietra. A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1), 1996.

[3] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

[4] O. Owoputi, B. O'Connor, C. Dyer, K. Gimpel, N. Schneider, and N. Smith. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of NAACL-HLT*, 2013.

[5] E.T.K. Sang and F. De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of HLT-NAACL*, 2003.

[6] J. Turian, L. Ratinov, and Y. Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of ACL*, 2010.