10 Leveraging Cross-Domain Social Media Analytics to Understand TV Topics Popularity

22 An Efficient Memetic Algorithm for Influence Maximization in Social Networks

34 Learning User and Product Distributed Representations Using a Sequence Model for Sentiment Analysis

45 Statistical Learning Theory and ELM for Big Social Data Analysis
In 2017, IEEE is going to bring out a new publication named IEEE Transactions on Emerging Topics in Computational Intelligence (TETCI). TETCI will publish original articles on emerging aspects of computational intelligence, including theories, applications, and surveys. This Transactions is sponsored by the IEEE Computational Intelligence Society and technically co-sponsored by the IEEE Computer Society. TETCI is an electronic only publication. The first issue will be published in February 2017.

**TETCI welcomes manuscripts in any emerging topic in computational intelligence, including but not limited to:**

- artificial life
- ambient intelligence
- brain computer interface
- computational neuroscience
- glial cell networks
- advanced linguistic computation
- cultural learning
- artificial endocrine networks
- social reasoning
- artificial hormone networks
- computational intelligence for IoT
- computational intelligence for Self-X and Smart-X technologies

For submission and other information, please visit TETCI website at: [http://cis.ieee.org/ieee-transactions-on-emerging-topics-in-computational-intelligence.html](http://cis.ieee.org/ieee-transactions-on-emerging-topics-in-computational-intelligence.html)
Features

10 Leveraging Cross-Domain Social Media Analytics to Understand TV Topics Popularity
by Ruggero G. Pensa, Maria Luisa Sapino, Claudio Schiavonella, and Luca Vignaroli

22 An Efficient Memetic Algorithm for Influence Maximization in Social Networks
by Maquo Gong, Chao Song, Chao Duan, Lijia Ma, and Bo Shen

34 Learning User and Product Distributed Representations Using a Sequence Model for Sentiment Analysis
by Tao Chen, Ruifeng Xu, Yulan He, Yunqing Xia, and Xuan Wang

45 Statistical Learning Theory and ELM for Big Social Data Analysis
by Luca Oneto, Frederica Bisio, Erik Cambria, and Davide Anguita

Columns

56 Research Frontier
Collaborative Brain–Computer Interface for People with Motor Disabilities
by Yiqing Li and Chang S. Nam

67 Discussion Forum
Human vs. Computer Go: Review and Prospect
by Chang-Shing Lee, Mei-Hui Wang, Shi-Jin Yuan, Ting-Han Wei, I-Chen Wu, Ping-Chiang Chou, Chun-Hsun Chou, Meng-Wai Wang, and Tai-Hsiung Yang

Departments

2 Editor’s Remarks
by Pablo A. Estévez

3 President’s Message
by Pablo A. Estévez

4 Conference Reports
Conference Report on 2015 IEEE Conference on Computational Intelligence and Games (IEEE CIG 2015)
by Chang-Shing Lee, I-Chen Wu, and Mei-Hui Wang

6 Publication Spotlight
by Haibo He, Chin-Teng Lin, Kay Chen Tan, Graham Kendall, and Yaohua Jin

8 Guest Editorial
Computational Intelligence for Big Social Data Analysis
by Erik Cambria, Newton Howard, Yunqing Xia, and Tat-Seng Chua

74 Conference Calendar
by Bernadette Bouchon-Meunier
Power of a Single Photo in the Big Data Era

In the current Big Data Era, an enormous amount of data has been automatically collected by various devices such as security cameras, sensor networks and automatic ticket gates with the advent of IoT. At the same time, billions of people are creating, sending and uploading a lot of information in various forms such as chats, photos and videos. With this daily information flood, it may be almost impossible for us to calculate the visible impact of uploading a single photo. For example, it is unlikely that my photo below at the Rusutsu Ski Resort will increase the number of foreign visitors for the next ski season.

However, on the contrary, there exists a number of surprising other stories. One well known story in Japan is about a photo of a young singer. Three years ago, she was a junior high-school student and a member of an unfamiliar local group of singers. A photo was taken by her fan at a small local event and uploaded to an online message board. Within a few days, the photo went viral and was widely shared. The photo of her had high visible attention. Now she is a popular singer and movie star. This success story shows that even a single photo can generate a large impact in our daily information flood. A similar story could happen with your WCCI paper amongst thousands of presentations. My favorite story is about a lost camera. An underwater camera was found on a beach of an island. A diving shop owner on the island took a photo of the camera and uploaded the photo as a lost-and-found item. Surprisingly, the owner of the camera was found in Taiwan within three days. The camera was lost in Taiwan ten months earlier and amazingly floated all the way to Japan.

The feature topic in the current CIM issue is “Big Social Data Analysis”. The proposal of this special issue was inspired by the great success of the two previous special issues on “Natural Language Processing” and “Big Data”. Those articles in the special issues have received a lot of attention. I hope that you will also enjoy all the articles in our current CIM issue.

Holiday at Rusutsu Resort in Hokkaido, Japan (February 2016).

Digital Object Identifier 10.1109/MCI.2016.2572461
Date of publication: 18 July 2016

Hisa0 Ishibuchi
Osaka Prefecture University, JAPAN
Frontiers of Knowledge in Computational Intelligence

Researchers move the frontiers of knowledge forward, generating new ideas and discoveries for the benefit of humanity. I have heard that expression, the “frontiers of knowledge,” for a long time, but only recently has it made real sense to me. Our research team is developing a pipeline for real-time processing of image streams coming from telescopes in northern Chile, with the aim of detecting supernovae, the explosive deaths of stars. Let us imagine a sphere of the Universe centered at the Earth, and let us call the surface of this sphere “the frontier”. The larger the radius of the sphere, the larger its surface and therefore the larger the number of galaxies and stars at the frontier. However in general terms, the larger the sphere radius, the fainter the light coming from stellar objects. A weaker signal means a lower signal-to-noise ratio (SNR), and a higher probability of non-detections or false discoveries. So, in order to push “the frontier” forward, we need to deal with the problem of reducing the rate of false discoveries at low SNRs. Note that we are moving the frontier backwards in time towards the origin of the Universe. Currently we are applying convolutional neural networks successfully to attack this problem. Now, every time that we reduce the rate of false discoveries and non-detections further, I have the strong feeling that we are pushing the “frontiers of knowledge” forward.

What are the frontiers of knowledge for your own research, and for the Computational Intelligence Society (CIS) community in general? The CIS community has been pushing the frontiers of knowledge forward in neural networks, evolutionary computation, and fuzzy systems for decades. But, hey, look, something new is appearing on the horizon—what are the emerging topics in Computational Intelligence (CI), and a space is needed for presenting new ideas and discoveries.

The good news is that we have already started creating a new journal, called *IEEE Transactions on Emerging Topics in Computational Intelligence* (TETCI), which is expected to be launched in February, 2017. TETCI will publish original articles on emerging aspects of computational intelligence, including theories, applications, and surveys. Authors are encouraged to submit manuscripts on any emerging topic in CI, especially nature-inspired computing topics not covered by other IEEE CIS journals. A few such examples by way of illustration are glial cell networks, computational neuroscience, brain computer interface, ambient intelligence, non-fuzzy computing with words, artificial life, cultural learning, artificial endocrine networks, social reasoning, artificial hormone networks, CI for the Internet-of-Things, and Smart-X technologies. Submission and review of manuscripts are now open through Manuscript Central, the on-line submission and review system used by IEEE. Interested researchers can also submit a proposal for a special issue on an emergent topic in CI. For more information, I invite you to have a look at the TETCI website: http://cis.ieee.org/ieee-transactions-on-emerging-topics-in-computational-intelligence.html.

(continued on page 9)
The annual IEEE Conference on
Computational Intelligence and
Games (IEEE CIG) is one of the
premier international conferences
in the field of computational intelligence
and games. The IEEE CIG 2015
(http://cig2015.nctu.edu.tw/) was held
at the Tayih Landis Hotel in Tainan,
Taiwan from August 31 to September
2, 2015. Sponsored by the IEEE Com-
putational Intelligence Society (CIS),
co-organized by National Chiao Tung
University (NCTU, Taiwan), National
University of Tainan (NUTN, Taiwan),
Chang Jung Christian University
(CJCU, Taiwan), National Dong Hwa
University (NDHU, Taiwan), and
Osaka Prefecture University (OPU,
Japan), IEEE CIG 2015 brought
together leading researchers and practi-
tioners from academia and industry to
discuss recent advances and explore
future directions in this field. For the
main technical track, we received a
total of 87 papers from 31 countries,
of which 55 papers were accepted for an
acceptance rate of 63%. The IEEE CIS
travel grants program provided travel
grants to assist three IEEE CIS student
members for presenting their papers.

The IEEE CIG 2015 program con-
stituted of four keynote speeches, three
tutorial sessions, two special sessions, 55
oral presentations, 11 poster presenta-
tions along with program demonstra-
tion, six competitions, and one board
game and curling game demonstration.
The technical program of IEEE CIG
2015 included the following:

- Four keynote speeches: Xin Yao
  (Co-evolutionary learning in game-
  playing), Simon Lucas (General
  video game AI: Challenges and
  applications), Martin Mueller (Com-
  puter Go research—The challenges
  ahead), and Graham Kendall (Where
  games meet hyper-heuristics).
- Three tutorials: Diego Perez
  (VGDL and the challenge of creat-
  ing agents for GVGP), Daniel
  Ashlock (Representations for evolu-
  tionary computation in games), and
  Risto Miikkulainen (Evolving neu-
  ral networks).
- Six competitions: The winners of
  competitions were determined by
  the competition organizers. Addi-
  tionally, the IEEE CIS also awarded
  the prizes to the top three ranked
  participants and top ranked student
  participants. Human vs. computer
  Go competition invited three Tai-
  wanese professional Go players

Digital Object Identifier 10.1109/MCI.2016.2572479
Date of publication: 18 July 2016

Participants attending the opening ceremony together with staff of IEEE CIG 2015 including Julian Togelius, Garrison Greenwood, Xin Yao, Simon Lucas, I-Chen Wu, NUTN President Tzung-Hsien Huang, and CJCU President Yung-Lung Lee (left 3-9 of the second line, respectively).
(Chun-Hsun Chou/9P, Kai-Hsin Chang/5P, and Li-Chun Yu/1P), three Taiwanese amateur Go players (Yung-Hao Hung/6D, Bo-Yuan Hsiao/6D, and Chi Chang/5D), and three computer Go programs (Zen/Japan, CGI/Taiwan, and Aya/Japan) to join the competition.

- Starcraft AI Competition: organized by Kyung-Joong Kim, Sehar Shahzad Farooq, In-Seok Oh, and Man-Je Kim;
- Fighting Game AI Competition: directed by Ruck Thawonmas;
- GVGAI Competition: organized by Diego Perez, Spyridon Samothrakis, Julian Togelius, Tom Schaul, and Simon Lucas;
- Geometry Friends Game AI Competition: organized by Rui Prada and Francisco Melo;
- Human vs. Computer Go Competition: co-organized by Taiwanese Association for Artificial Intelligence (TAAI) and NUTN;
- U-Generation Islands of Adventure (E-game) Competition: co-organized by Information Education Center, Bureau of Education, Kaohsiung City Government, Taiwan.

The welcome reception, IEEE Member Activities (chaired by Pau-Choo Julia Chung), and IEEE CIS Game TC Meeting (chaired by Mike Preuss) were well attended on August 31. The IEEE TCIAIG AE Meeting (chaired by Graham Kendall) and the conference banquet were also organized by the IEEE CIG 2015 on September 1. In addition, the following awards were presented at the conference.

- Special Appreciation Award presented to keynote speakers, tutorial speakers, professional Go players, and developer of E-game competition;

Finally, we are looking forward to seeing all of you again at IEEE CIG 2016 held in Santorini, Greece on September 20-23, 2016.
IEEE Transactions on Neural Networks and Learning Systems


Digital Object Identifier: 10.1109/TNNLS.2015.2496330

“This paper proposes a generalized prediction system called a recurrent self-evolving fuzzy neural network (RSEFNN) that employs an on-line gradient descent learning rule to address the electroencephalography (EEG) regression problem in brain dynamics for driving fatigue. The cognitive states of drivers significantly affect driving safety; in particular, fatigue driving, or drowsy driving, endangers both the individual and the public. For this reason, the development of brain–computer interfaces (BCIs) that can identify drowsy driving states is a crucial and urgent topic of study. Many EEG-based BCIs have been developed as artificial auxiliary systems for use in various practical applications because of the benefits of measuring EEG signals. However, the efficacy of EEG-based BCIs in recognition tasks has been limited by low resolutions. The system proposed in this paper represents the first attempt to use the recurrent fuzzy neural network (RFNN) architecture to increase adaptability in realistic EEG applications to overcome this bottleneck. This paper further analyzes brain dynamics in a simulated car driving task in a virtual-reality environment. The proposed RSEFNN model is evaluated using the generalized cross-subject approach, and the results indicate that the RSEFNN is superior to competing models regardless of the use of recurrent or nonrecurrent structures.”

IEEE Transactions on Fuzzy Systems


Digital Object Identifier: 10.1109/TFUZZ.2015.2426212


Digital Object Identifier: 10.1109/TNNLS.2015.2481006

“This paper presents several L₁-minimization algorithms for sparse signal reconstruction based on a continuous-time projection neural network (PNN). First, a one-layer projection neural network is designed based on a projection operator and a projection matrix. The stability and global convergence of the proposed neural network are proved. Then, based on a discrete-time version of the PNN, several L₁-minimization algorithms for sparse signal reconstruction are developed and analyzed. Experimental results based on random Gaussian sparse signals show the effectiveness and performance of the proposed algorithms. Moreover, experimental results based on two face image databases are presented that reveal the influence of sparsity to the recognition rate. The algorithms are shown to be robust to the amplitude and sparsity level of signals as well as efficient with high convergence rate compared with several existing L₁-minimization algorithms.”
“In recent years, most fuzzy system software has been developed in order to facilitate the use of fuzzy systems. Some software is commercially distributed, but most software is available as free and open-source software, reducing such obstacles and providing many advantages: quicker detection of errors, innovative applications, faster adoption of fuzzy systems, etc. In this paper, the authors present an overview of freely available and open-source fuzzy systems software in order to provide a well-established framework that helps researchers to find existing proposals easily and to develop well-founded future work. To accomplish this, they propose a two-level taxonomy, and describe the main contributions related to each field. Moreover, they provide a snapshot of the status of the publications in this field according to the ISI Web of Knowledge. Finally, some considerations regarding recent trends and potential research directions are presented.”

Digital Object Identifier: 10.1109/TFUZZ.2015.2428717

“This study elaborates on a comprehensive design methodology of fuzzy cognitive maps (FCMs). In this paper, the maps are regarded as a modeling vehicle of time series. It is apparent that whereas time series are predominantly numeric, FCMs are abstract constructs operating at the level of abstract entities referred to as concepts and represented by the individual nodes of the map. The authors introduce a mechanism to represent a numeric time series in terms of information granules constructed in the space of amplitude and change of amplitude of the time series, which, in turn, gives rise to a collection of concepts forming the corresponding nodes of the FCMs. Each information granule is mapped onto a node (concept) of the map. The learning is typically realized in a supervised mode on a basis of some experimental data. The proposed approach is illustrated in detail by a series of experiments using a collection of publicly available data.”

Digital Object Identifier: 10.1109/TEVC.2015.2428292

“The performance of evolutionary algorithms can be heavily undermined when constraints limit the feasible areas of the search space. For instance, while covariance matrix adaptation evolution strategy (CMA-ES) is one of the most efficient algorithms for unconstrained optimization problems, it cannot be readily applied to constrained ones. Here, the authors used concepts from memetic computing, i.e., the harmonious combination of multiple units of algorithmic information, and viability evolution, an alternative abstraction of artificial evolution, to devise a novel approach for solving optimization problems with inequality constraints. Viability evolution emphasizes the elimination of solutions that do not satisfy viability criteria, which are defined as boundaries on objectives and constraints. These boundaries are adapted during the search to drive a population of local search units, based on CMA-ES, toward feasible regions. These units can be recombined by means of differential evolution operators. Of crucial importance for the performance of the method, an adaptive scheduler toggles between exploitation and exploration by selecting to advance one of the local search units and/or recombine them. The proposed algorithm can outperform several state-of-the-art methods on a diverse set of benchmark and engineering problems, both for quality of solutions and computational resources needed.”

Digital Object Identifier: 10.1109/TEVC.2015.2443057

“Bilevel optimization problems are characterized by a hierarchical leader-follower structure, in which the leader desires to optimize her own strategy taking the response of the follower into account. These problems are referred to as Stackelberg problems in the domain of game theory, and as bilevel problems in the domain of mathematical programming. In a number of practical scenarios, a bilevel problem is solved by a leader who needs to take multiple objectives into account and simultaneously deal with the decision uncertainty involved in modeling the follower’s behavior. Such problems are often encountered in strategic product design, homeland security applications, and taxation policy. However, the hierarchical nature makes the problem difficult to solve and they are commonly simplified by assuming a deterministic setup with smooth objective functions. In this paper, the authors focus on the development of a flexible evolutionary algorithm for solving multicriteria bilevel problems with lower level (follower) decision uncertainty. The performance of the algorithm is evaluated in a comparative study on a number of test problems. In addition to the numerical experiments, the authors consider two real-world examples from the field of environmental economics and management to illustrate how the framework can be used to obtain optimal strategies.”

Digital Object Identifier: 10.1109/TCIAIG.2014.2346242

(continued on page 75)
in the eras of social connectedness and social colonization, people are becoming increasingly enthusiastic about interacting, sharing, and collaborating through online collaborative media. In recent years, this collective intelligence has spread to many different areas, with particular focus on fields related to everyday life such as commerce, tourism, education, and health, causing the size of the Social Web to expand exponentially. The distillation of knowledge from such a large amount of unstructured information, however, is an extremely difficult task, as the contents of today's Web are perfectly suitable for human consumption, but remain hardly understandable to machines.

Big social data analysis grows out of this need and combines multiple disciplines such as social network analysis, multimedia management, social media analytics, trend discovery, and opinion mining. For example, studying the evolution of a social network merely as a graph is very limited as it does not take into account the information flowing between network nodes. Similarly, processing social interaction contents between network members without taking into account connections between them is limited by the fact that information flows cannot be properly weighted. Big social data analysis, instead, aims to study large-scale Web phenomena such as social networks from a holistic point of view, i.e., by concurrently taking into account all the socio-technical aspects involved in their dynamic evolution. Hence, big social data analysis is inherently interdisciplinary and spans areas such as machine learning, graph mining, information retrieval, knowledge-based systems, linguistics, common-sense reasoning, natural language processing, and big data computing.

Big social data analysis finds applications in several different scenarios. There is a good number of companies, large and small, that include the analysis of social data as part of their missions. Big social data analysis can be exploited for the creation and automated upkeep of review and opinion aggregation websites, in which opinionated text and videos are continuously gathered from the Web and not restricted only to product reviews, but also to wider topics such as political issues and brand perception. Big social data analysis has also a great potential as a sub-component technology for other systems. They can enhance the capabilities of customer relationship management and recommendation systems allowing, for example, to find out which features customers are particularly happy about or to exclude the recommendations items that have received very negative feedbacks. Similarly, they can be exploited for affective tutoring and affective entertainment or for troll filtering and spam detection in online social communication.

Business intelligence is also one of the main factors behind corporate interest in big social data analysis. Nowadays, companies invest an increasing amount of money in marketing strategies and they are constantly interested in both collecting and predicting the attitudes of the general public towards their products and brands. The design of automatic tools capable to mine sentiments over the Web in real-time and to create condensed versions of them represents one of the most active research and development areas. The development of such systems, moreover, is not only important for commercial purposes, but also for government intelligence applications able to monitor increases in hostile communications or to model cyber-issue diffusion.

Several commercial and academic tools track public viewpoints on a large-scale by offering graphical summarizations of trends and opinions in the blogosphere. Nevertheless, most commercial off-the-shelf (COTS) tools are limited to a polarity evaluation or a mood classification according to a very limited set of emotions. In addition, such methods mainly rely on parts of text in which emotional states are explicitly expressed and, hence, they are unable to capture opinions and sentiments that are expressed implicitly. Because they are mainly based on statistical properties associated with words, in fact, many COTS tools are easily tricked by linguistic operators such as negation and disjunction.

The main motivation for this Special Issue is to explore how computational intelligence can help overcome such hurdles through new forms of processing that can handle high volume, high
velocity, and high variety information assets and, hence, enable a more efficient passage from (unstructured) social information to (structured) machine-processing data, in potentially any domain. We received 34 submissions, which were reviewed by 63 prominent scholars, and selected 4 articles out of them.

The article "Leveraging Cross-Domain Social Media Analytics to Understand TV Topics Popularity" by Ruggero Pensa, Maria Luisa Sapino, Claudio Schifanella, and Luca Vignaroli proposes a concept-level integration framework in which users’ activities on different social media are collectively represented, and possibly enriched with external knowledge, such as information extracted from the Electronic Program Guides, or available ontological domain knowledge. The integration framework has a knowledge graph as its core data model. It keeps track of active users, the television events they talk about, the concepts they mention in their activities, as well as different relationships existing among them. Temporal relationships are also captured to enable temporal analysis of the observed activity. The data model allows different types of analysis and the definition of global metrics in which the activity on different media concurs with the measure of success.

In "An Efficient Memetic Algorithm for Influence Maximization in Social Networks", Maoguo Gong, Chao Song, Chao Duan, Lijia Ma, and Bo Shen implement a memetic algorithm for community-based influence maximization in social networks. Such an algorithm optimizes the 2-hop influence spread to find the most influential nodes. Problem-specific population initialization and similarity-based local search are designed to accelerate the convergence of the algorithm. Experiments on three real-world datasets demonstrate that the proposed algorithm has competitive performance to the compared algorithms in terms of effectiveness and efficiency. For example, on a real-world network of 15233 nodes and 58891 edges, the influence spread of the proposed algorithm is 12.5%, 13.2% and 173.5% higher than the three frequently used algorithms Degree, PageRank and Random, respectively.

Next, in “Learning User and Product Distributed Representations Using a Sequence Model for Sentiment Analysis”, handled independently during review process, Tao Chen, Ruifeng Xu, Yulan He, Yunqing Xia, and Xuan Wang argue that the temporal relations of product reviews might be potentially useful for learning user and product embedding and thus propose employing a sequence model to embed these temporal relations into user and product representations so as to improve the performance of document-level sentiment analysis. Specifically, a distributed representation of each review is first learnt by a one-dimensional convolutional neural network. Then, taking these representations as pre-trained vectors, authors use a recurrent neural network with gated recurrent units to learn distributed representations of users and products. Finally, user, product and review representations are fed into a machine learning classifier for sentiment classification. Experimental results show that sequence modeling for the purposes of distributed user and product representation learning can improve the performance of document-level sentiment classification.

Finally, “Statistical Learning Theory and ELM for Big Social Data Analysis” is presented by Luca Oneto, Federica Bisio, Erik Cambria, and Davide Anguita. The paper, handled independently during review process, illustrates how to exploit the most recent technological tools and advances in Statistical Learning Theory in order to efficiently build an Extreme Learning Machine (ELM) for big social data analysis. ELM represents a powerful learning tool, developed to overcome some issues in back-propagation networks. The main problem with ELM is that it trains it to work in the event of a large number of available samples, where the generalization performance has to be carefully assessed. For this reason, the authors propose an ELM implementation that exploits the Spark distributed in memory technology in order to address the issue of selecting ELM hyperparameters that give the best generalization performance.
Leveraging Cross-Domain Social Media Analytics to Understand TV Topics Popularity

Abstract—The way we watch television is changing with the introduction of attractive Web activities that move users away from TV to other media. The social multimedia and user-generated contents are dramatically changing all phases of the value chain of contents (production, distribution and consumption). We propose a concept-level integration framework in which users’ activities on different social media are collectively represented, and possibly enriched with external knowledge, such as information extracted from the Electronic Program Guides, or available ontological domain knowledge. The integration framework has a knowledge graph as its core data model. It keeps track of active users, the television events they talk about, the concepts they mention in their activities, as well as different relationships existing among them. Temporal relationships are also captured to enable temporal analysis of the observed activity. The data model allows different types of analysis and the definition of global metrics in which the activity on different media concurs with the measure of success.

I. Introduction

The introduction of attractive Web activities that move users away from television to other media is changing the way we watch TV. Also the media broadcasting models are changing in order to cover the TV-Web convergence. VoD (Video on Demand) and EPGs (Electronic Program Guides) provided by broadcasters are examples of services that allow new forms of user navigation within the television content. At the same time, the popularity of online social networks has changed the Internet ecosystem, thus leading to more collaborative environments, reflecting the structure and dynamics of the society. The social multimedia and user-generated contents are dramatically changing all phases of the value chain of contents (production, distribution and consumption).

For the broadcasters and advertisers, social TV means much deeper real-time understanding of what viewers think about shows and the brands that advertise on them. Consequently, it provides them with data-driven understanding of their investments in contents. This will be the most significant change that social TV brings to the TV business. Both the commissioning and scheduling of TV contents and the

Ruggero G. Pensa and Maria Luisa Sapino
Department of Computer Science,
University of Torino, Torino, ITALY

Claudio Schifanella and Luca Vignaroli
RAI-CRIT, Torino, ITALY

Digital Object Identifier 10.1109/MCI.2016.2572518
Date of publication: 18 July 2016

Corresponding Author: Ruggero G. Pensa (Email: ruggero.pensa@unito.it).
pricing of the spot ads and program sponsorship are based on the way the TV audience is measured. There is no doubt that traditional TV ratings still rank as an important measurement that advertisers pay attention to when buying advertising inventory; however, it is possible that in the upcoming years social TV data will shift attention away from traditional audience ratings.

Traditionally, measures about people’s habits and reactions are gathered in two ways: firstly, by viewing habits of panels of TV viewers and parsing the results of network surveys on the opinion (e.g., the Nielsen ratings); secondly, by generating traditional live broadcast audience figures with the so-called set meters (small devices connected to TVs in a small number of selected homes) on a daily basis. However, this approach misses the explosive growth and increasing diversity of comments and opinions in real time from an expanding number of online social platforms. In the typical offline scenario, audience profiles are obtained manually by gathering a set of predefined socio-demographics characteristics obtained from a statistically significant sample of the possible consumers. By contrast, in an online scenario, audience profiles and impact of TV programs might be obtained by tracking social media sites (e.g., Twitter, Facebook) and applying Natural Language Processing (NLP) technologies and data mining techniques on their contents. This might enable TV reprogramming and media planning strategies, such as contextual advertisement or behavioral targeting.

In this paper, we observe that, although different social media are characterized by different users’ activity styles, they all carry useful information (i.e., non-redundant with respect to each other). While Twitter activities have the peculiarity of being very timely and immediate—usually users tweet in real time, while watching the program they are commenting about—a good portion of the activities on YouTube and Facebook happens with some time shift with respect to the on-air show. Users post fragments of videos, which potentially trigger comments and discussions for days, in some cases even weeks or months. Thus, we propose a concept-level integration framework in which users’ activities on different social media are collectively represented by means of conceptual abstractions, possibly enriched with external knowledge, such as information extracted from the EPGs, or available ontological domain knowledge.

The framework has a knowledge graph as its core data model, which keeps track of active users, the television events they talk about, the concepts they mention in their activities, as well as different relationships existing among them, including temporal relationships which enable temporal analysis of the observed activity. The data model allows different types of analysis and the definition of global metrics in which the activity on different media concurs with the measure of success.

Note that, although we concentrate our cross-media analysis on the study of the popularity of topics tackled in television programs, we do believe that the concept-level integration platform has the property of being very general. As such, it has the potential of being populated and enriched with information of interest in different domains (such as tracking political dynamics, tracking the correlation between users’ social activities and economic patterns). Thus, in the rest of the paper, we first define the general concept, and then show how they are instantiated in the TV domain.

The paper is organized as follows: after presenting a survey of related literature in Section II, we introduce our integration framework in Section III. We formally define the graph integration
The introduction of attractive Web activities [...] is changing the way we watch TV. [...] In the upcoming years social TV data will shift attention away from traditional audience ratings.

model in Section IV, and describe the source processing steps which extract the concepts and relationships that will populate the graph in Section V. In Section VI, we define formally some queries of interest. Section VII shows the potential of the graph as a data source to analyze topics’ popularity. Finally, we draw the conclusion in Section VIII.

II. Related Work
In this section, we will provide an overview of relevant related work. As the main focus of this paper is on cross-network analysis to capture the interest users show towards topics addressed in TV programs, we will first survey literature on cross-network analysis. Then we will discuss graph based multi-source data integration, which relates to our paper in that our analysis relies on a graph data model to represent multi-source social media information.

A. Cross-Network Analysis
Social network analysis has recently been a core method to understand various phenomena potentially influenced by the exchange of users’ opinions. Retrieving information from social media is then a crucial preliminary task. At this purpose, in [1], the authors investigate how to automatically retrieve a context-relevant social network content without user intervention, by considering both the participatory and implicit-topical properties of the context to improve the retrieval performance. Users’ activities on Twitter are used to support the prediction of economic phenomena, such as stock prices [2], and for tracking online social movements [3]. Kaschesky et al. [4] use sentiment analysis to predict the political orientation of a person (Republican vs. Democrat) or agreement/disagreement on political issues. Alashri et al. [5] analyze online ideological political debates, defined as a formal discussion on a set of related issues in which opposing perspectives and arguments are put forward.

Recently TV broadcasters recognized that users’ activities on social media are valuable sources of information about their interests towards TV programs. In 2012, Nielsen—a global information and measurement company—and Twitter agreed to create the “Nielsen Twitter TV Rating” for the US market. The main goal of their agreement is the definition of a metric relying on conversations about TV programs on Twitter to measure users’ interests. The metric provides valuable information for TV contents recommendation, including personalized commercial campaigns. In the context of TV and social Web integration, Bluefin Labs1 releases a suite of analytics tools to explore the social content related to Social TV programs and to analyze the data generated by the “TV Genome”, i.e., the mapping between social media and TV media. This software is based in large part on researches on natural language processing, speech-to-text and video-entity recognition carried out by the two co-founders [6], [7].

B. Personalization and Recommendation in the TV Domain
The study [8] introduces a linear time algorithm to solve the problem that involves selecting different program slots telecast on different television channels in a day so as to reach the maximum number of viewers. O’Sullivan et al. [9] address the problem of creating personalized EPGs in the digital TV domain by applying data mining methods to extract new program metadata from user profiles. Yan et al. [10] propose a YouTube video recommendation solution via cross-network collaboration: the authors concentrate on those users who are active both on Twitter and on YouTube, and exploit the users’ profile information that they can learn by analyzing their activity on Twitter to personalize YouTube video recommendations. While being similar to our approach for the basic idea of integrating information coming from different social media, the work presented in [10] significantly differs from the cross-network concept-level integration proposed in this paper.

C. Graph Based Knowledge Representation, Integration and Querying
The necessity of structuring knowledge in a graph was already identified in 1988 by [11] as a means of representing knowledge from multiple sources in knowledge-based systems. Ten years after, this necessity has been translated into the design of information storage and retrieval systems such as the one presented by [12]. Today, knowledge graphs are exploited by semantic analysis [13], sentiment analysis [14] and opinion mining [15]. Furthermore, time is also a key question in knowledge representation and analysis [16]. As an example, Google uses a knowledge graph for its search engine.

In more recent years, many researchers have focused their efforts in identifying a way to represent heterogeneous, multimedia and multi-language ontological knowledge embracing a wide range of domains. For instance, the studies [17], [18] introduce Freebase, a tuple database used to structure general human knowledge. Navigli and Ponzetto [19] present an automatic approach to the construction of BabelNet, a very large, wide-coverage multilingual semantic network by integrating lexicographic and encyclopedic knowledge from WordNet and Wikipedia.

Querying and analyzing these knowledge graphs is a key issue in heterogeneous knowledge-based systems. The paper [20] presents an abstract machine dedicated to querying knowledge graphs as the result of an abstraction process performed to reach a generic solution to the problem of querying graphs in various models. The authors of [21] present a web-based system for visual and interactive analysis of large sets of documents using statistical topic models. This work proposes a range of

1http://bluefinlabs.com/
visualization types and control mechanisms to support knowledge discovery, including corpus and document specific views, iterative topic modeling, search, and visual filtering.

As graphs become increasingly large, scalability quickly becomes the major research challenge for the reachability computation today. Many works propose different indices to answer reachability queries efficiently [22]–[24]. Jin et al. [25] propose a unified reachability computation framework scaling reachability indices to help speed up the online query answering approaches. When knowledge graphs become huge, the relevance of the returned results is a key issue at least as the response time. The study [26] addresses the problem of an index structure through the design and implementation of a concept-based model using domain-dependent ontologies.

Our work is transverse to the presented related researches. Not only do we provide a theoretical framework for concept-level heterogeneous and time-evolving data integration, management and querying, but we also develop a web-based application guiding the user in the exploration, analysis and visualization of the complex and dynamic interactions constituting the “extended life” of TV events.

A preliminary version of this work has been published in [27], [28]. This paper significantly extends our former publications by adding many previously missing technical details. In particular, we now provide the formal definitions of the domain-specific model as well as the theoretical foundations of our querying framework. Finally, in this work we report the results of an experiment conducted on a more recent and large-scale scenario.

III. The integration Framework

Our cross-network analysis framework was first introduced in [27] and consists of three main layers, covering all the phases from data collection, representation and integration to data analysis. More specifically, a source processing layer contains the different modules for collecting all the data to be conveyed in the knowledge representation model. It accesses a number of predefined web/social/media sources (e.g., broadcasters official web sites, social networks, TV channels, ontological information sources) and extracts from them those information units which denote relevant concepts (e.g., people names, geographical names, temporal information, topic names, etc.) as well as information supporting the existence of relationships (which will be modeled as edges in the graph) among them.

The collected concepts and relationships among them will be organized in a structured knowledge graph by the knowledge graph layer, which contains all the modules needed to define and store the knowledge graph.

The knowledge query and analysis layer offers functionalities for querying, browsing and analyzing the knowledge graph. More specifically, a query module extracts subgraphs from the knowledge graph based on user’s requirements and constraints. Each extracted subgraph can be seen as a “view” over the complete knowledge graph, only containing nodes and edges potentially relevant to the user query. An analysis module provides a set of analysis and data mining components to extract models and patterns from the knowledge graph. Both the entire knowledge graph, and the individual views (subgraphs extracted from the query module) can be subject to data analysis. Tensor based representations [29] are also provided to enable the direct application of existing matrix and tensor-based analysis libraries, as well as the definition of innovative analysis algorithms efficiently dealing with the multidimensional characteristics of the modeled knowledge.

Notice that in our integration framework a fundamental role is played by a semantic engine. First, it is adopted in the source processing layer to provide an interpretation to web/social/media elements extracted from heterogeneous sources. In the source processing phase, the semantic engine helps understand whether the considered entities should be modeled as concepts or relationships among existing concepts, and helps provide a suitable set of features based on their characteristics. Second, the semantic engine plays an important role in the graph query and analysis layer, where it assigns a semantic role to each selected node/edge.

In the following sections, we describe the three phases of the frameworks in details. We first present in details the definition of the knowledge graph, which is the core of our proposal. Then we discuss how the data sources are processed to extract the relevant information and populate the knowledge graph. Finally, we define the formal query and analysis framework.

IV. Modeling Cross-Network Knowledge

The core of our framework is the knowledge base that represents the result of public actions of users in social environments [27], [28]. Combining different theories from cognitive science [30], [31], language philosophy [32] and social ontology [33], we recognize three classes of entities, which will be mapped into three types of nodes in the knowledge graph: subjects, i.e., users who take public actions (such as posting a tweet), social objects, i.e., the result of public acts (such as a set of tweets posted by a user), and concepts, physical and/or ideal objects mentioned by subjects via their public actions. Any act (or set of acts) that can be identified by its trace, and has a recognized social value is a social object. Given the size of the domain of interest, and the granularity of the analysis we are interested in, in this paper we choose to model social objects to represent groups of similar actions instead of keeping track of the individual subjects’ actions. This assumption could be relaxed if we were interested in distinguishing every single users’ action (for example, to work towards personalized recommendation systems).
We capture different existing relationships between subjects and social objects, and between social objects and concepts: a group of subjects that recognize a social value of an act supports the resulting social object (e.g. the contractors support the contract); a social object represents a social instance of some concepts on a precise context (e.g. a video may represent a volleyball match). Other relationships exist among entities of the same type. We call these relationships structural dependencies. A social object \( o_1 \) is structurally dependent on another object \( o_2 \) if \( o_1 \) is a part of \( o_2 \) (e.g. a comment is a part of a video). A subject can be structurally dependent on a group of subjects (e.g. a subscriber is a part of playlist subscribers) that performed the same kind of actions on the same social object. A concept may be structurally dependent on a more general concept (e.g. hiliarity is a specialization of joy).

Finally, we capture the fact that social objects evolve with time. Hence, as a special case of representation relationship, we consider the temporal relation between a social object and a temporal concept (e.g. a video has been posted in a specific time instant, and has been viewed during a specific time period).

Based on the above, in the following subsection, we formally define the knowledge graph.

**A. Knowledge Graph**

The knowledge graph (first introduced in [27]) models all the relationships between social objects, subjects and concepts introduced so far.

**Definition 1 (Knowledge Graph):** Let \( O, S \) and \( C \) be the sets of all social objects, subjects and concepts, respectively. Let \( T \subseteq C \) be the set of temporal concepts. The cross-network knowledge graph on \( O, S, \) and \( C \) is the directed weighted graph \( G^k(V, E, W) \), where the set of vertices is \( V = O \cup S \cup C \) and the set of edges is \( E = E^w \cup E^{rep} \cup E^{str} \) including edges representing support relationships, representation relationships as well as structural dependency relationships. In particular \( E^{rep} = \{(o, c) : s.t. \; o \in O, c \in C \} \) is the set of representation edges, and \( E^{str} = \{(v,v') : s.t. \; v,v' \in S \lor v,v \in O \lor v,v' \in C, i \neq j \} \) is the set of structural dependency edges. The edge weighting function is \( W: E \rightarrow \{0, 1\} \).

Each node \( v \in V \) has three attributes: \( v\_{label}, v\_{subtype} \) and \( v\_{magnitude} \), representing the name of the concept associated to the node, an application-specific type and the number of instances of such concept recognized in the data sources, respectively. Optionally, each edge \( e \in E \) may be characterized by an attribute \( e\_{subtype} \) which specifies an application-specific type. Moreover, given the set of time concepts \( T \subseteq C \), \( E^{temp} \subseteq E^{str} \) denotes the set of edges \( \{(o, t) \} \) where \( o \in O \), and \( t \in T \).

A special subgraph of \( G^k \) is the ontology graph. Vertices in the knowledge base are all concepts belonging to \( C \), defined as follows.

**Definition 2 (Ontology Graph):** The ontology graph \( G^O(V^O, E^O, W^O) \) is the subgraph of \( G^k \) induced by \( V^O = C \).

Thus, \( E^O \) is a set of structural dependency edges encoding several ontology relationships (such as: “is a”, “part of”), possibly specified by the attribute \( e\_{subtype} \).

Fig. 1 (left side) shows a small example of knowledge graph in which every type of node and edge is represented.

**V. Extracting Information from Social Media Data Sources**

In this section, we describe how the user-generated content publicly available on social media is processed and mapped to the concepts and relationships to be represented in the knowledge graph defined in Section IV.

**A. Facebook Content**

Facebook is the most famous and widespread social networking platform. Since it has about 1.5 billions of monthly active users, TV companies use it for stimulating discussions around TV shows. Through the website, users may post, watch, comment, like or dislike any kind of multimedia comments: text comments,
The term “concept” refers both to an abstraction and to an instantiation of an abstract concept. This choice supports the necessity of considering the evolution of users’ perception of concepts in time.

**Definition 3 (commentset):** Given a time interval $\Delta T = (t_{\text{start}}, t_{\text{end}})$ and a Facebook post $f$, a commentset $CS(\Delta T, f)$ is a collection of comments posted during $\Delta T$ by users as a reaction to post $f$.

We store posts that are closely related to a TV event, e.g., because they are published by the broadcaster’s official channel/user or by other users mentioning the TV event in the title or in the description. Here, we define as TV event an individual episode of a television series (e.g., news programs, fiction series, magazines) or a single production (e.g., a football match, a film, a live concert).

In particular, for each Facebook post $f$ we store: i) a social object node $o_\text{s}$ representing the Facebook post; ii) a representation edge $e_\text{sac}$ linking $o_\text{s}$ to $c_\text{e}$ (the node corresponding to the target TV event $o_\text{e}$); iii) a set $N$ of social object nodes $o_\text{e}$ representing the N commentsets $CS(\Delta T, f)$, $j = 1, \ldots, N$; iv) the attributes $o_\text{s}.\text{magnitude}$ that represent the number of comments for $f$ in $CS(\Delta T, f)$; v) two time nodes $t_s$ and $t_e$ for each commentset node $o_\text{e}$ with $t_.label = t_{\text{start}}$ and $t_.label = t_{\text{end}}$; vi) two edges $e_{\text{sac}}$ and $e_{\text{sac}}'$ connecting each $o_\text{s}$ to $t_s$ and $t_e$.

Instead of storing the full textual content of the comments associated to the commentset, we only consider the most relevant concepts referred by the text and belonging to the following categories: people, places, events, emotions, polarities. We refer to these concepts as named entities. In particular, for each category, we store: i) a social object node $o_\text{e}$ representing the category; ii) a concept node $c_\text{e}$ for each relevant concept referred by $CS(\Delta T, f)$ and related to the category of $o_\text{e}$; iii) an edge $e_{\text{eac}}$ connecting each concept node $c_\text{e}$ to $o_\text{s}$, and a weight $w_{c_\text{e}}$ that expresses the relative importance of the concept $c_\text{e}$ (note that $\sum w_{c_\text{e}} = 1$); iv) a structure edge $e_{\text{sac}}$ that connects $o_\text{e}$ to $o_\text{s}$, and a weight $w_{c_\text{e}}$ expressing the relative importance of the category $o_\text{e}$ w.r.t. the other categories (note that $\sum w_{c_\text{e}} = 1$); v) the attribute $o_\text{e}.\text{magnitude}$ that expresses the total occurrences of all concepts connected to $o_\text{e}$.

Notice that, in this work, we refer to the word concept both as an abstraction (e.g., person, tv show, event) and as an instantiation of an abstract concept (e.g., “Barack Obama”, “Late Show”, “Rio 2016 Olympic Games”). This choice is justified by the necessity of considering the evolution of users’ perception of persons, events, places in time. According to our assumption, for instance, “Arnold Schwarzenegger” may be considered as a general concept that can be associated to his career as an actor or as a politician.

People who have contributed to the creation of social objects are represented as well. As for concepts, we include in the model only the most representative users belonging to the following categories: viewers, active users, uploaders, and broadcasters. In details, for each user category, we store: i) a subject node $s_\text{v}$ representing the user category; ii) a subject node $s_\text{u}$ for each relevant user extracted from $CS(\Delta T, f)$ and related to the category referred by $s_\text{v}$; iii) a structure edge $e_{\text{suc}}$ connecting each subject node $s_\text{u}$ to $s_\text{v}$, and a weight $w_{s_\text{u}}$ that expresses the relative importance of the user $s_\text{u}$ in the considered category (note that $\sum w_{s_\text{u}} = 1$); iv) a support edge $e_{\text{sac}}$ that connects $s_\text{u}$ to $o_\text{s}$, and a weight $w_{s_\text{u}}$ expressing the relative importance of the category $s_\text{v}$ w.r.t. the other categories (note that $\sum w_{s_\text{u}} = 1$); v) the attribute $s_\text{v}.\text{magnitude}$ that expresses the total occurrences of all concepts connected to $s_\text{v}$.

It is worth noting that YouTube videos can be processed as Facebook posts.

**B. Twitter Content**

Twitter is certainly one of the most dynamic social networking platforms due to its well-known peculiarities (among others, Twitter posts—called tweets—are limited to 140 characters and can be sent from any mobile device). Any kind of live events is often followed by thousands of tweets, thus providing a huge source of information for the analysts. TV programs’ editorial boards are used to propose specific topics of discussions by using hashtags triggering huge amounts of new tweets. Consequently, Twitter is often adopted as the preferred means to let the audience express instant feelings and opinions about what is being broadcasted. As such, it is a key source of information for our application. However, unlike Facebook that has a clear social unit (the post) corresponding to a knowledge graph node, a social node in Twitter is harder to identify. We may think about creating a node for each tweet, but it has two main drawbacks: (i) it is often semantically poor and (ii) the knowledge graph is subject to a rapid explosion. Another possibility is to identify a prolific Twitter user as a central social node. However, this choice is questionable too: a Twitter user is similar to a Facebook user. It provides subjective representations of the reality. To cope with this issue, we define a new social entity called tweetset, defined as follows:

**Definition 4 (tweetset):** Given a time interval $\Delta T = (t_{\text{start}}, t_{\text{end}})$ and a TV event $e$, a tweetset $TS(\Delta T, e)$ is a collection of tweets posted during $\Delta T$ and closely related to $e$.

For instance, the retained tweets are those mentioning hashtags and posted by users associated to a specific TV event (e.g., the official Twitter users/hashtags). As such, the portion of knowledge graph corresponding to a tweetset $TS(\Delta T, e)$ is given by: i) a social object node $o_\text{s}$ representing the tweetset; ii) a representation edge $e_{\text{sac}}$ connecting node $o_\text{s}$ to node $c_\text{e}$, corresponding to the TV event $e$; iii) the node attribute $o_\text{s}.\text{magnitude}$, initialized with the number of tweets in $TS(\Delta T, e)$; iv) two
The knowledge graph represents huge amounts of rich, heterogeneous and time-evolving information. Querying the graph in a simple but efficient way is then crucial for the usability of the system.

time nodes $t_1$ and $t_2$, with $t_1.label = t_{max}$ and $t_2.label = t_{end}$; v) two edges $e^{wp}$ and $e^{rn}$ connecting $o_i$ to $t_1$ and $t_2$.

Finally, we consider the concepts referred by the text in the tweetsets and the most relevant users. Both concepts and users are stored in the same way as described in Section V-A.

C. Enriching Social Media Information with Knowledge from the EPG Content

The official EPG is often provided as a static content by broadcasters themselves. Although this makes us think about EPGs as non-social content, in our application this assumption is false. In fact, EPGs may be enriched by information coming from the social platform that provides users’ rating, reviews, descriptions, and so on. Including these social sources of information is required when the official EPG content is poor. We consider a central social node corresponding to the TV event, the unit of an EPG schedule. This node is connected on one side to concepts identifying persons, places and events referred by each TV event, on the other side to concepts related to the TV event itself and its related TV program. In detail, the portion of the knowledge graph related to a TV event broadcasted from time $t_{start}$ to time $t_{end}$ is structured as follows: i) a social object node $o_i$ representing the TV event; ii) a representation edge $e^{rt}$ connecting node $o_i$ to node $e_i$, the conceptual node corresponding to the TV event; iii) two time nodes $t_1$ and $t_2$ with $t_1.label = t_{start}$ and $t_2.label = t_{end}$; iv) two edges $e^{wp}$ and $e^{rn}$ connecting $o_i$ to $t_1$ and $t_2$.

Finally, we consider some concepts referred by EPG sources in the following categories: people, places, events, genre and tv channel. We refer to Section V-A for a detailed description of how these concepts are stored.

D. Enriching Social Media Information with Domain Onologies

In addition to the social part of the knowledge graph, we consider other sources of knowledge that form the subgraph $G^O$ of $G^K$. In particular, we import ontology nodes from DBPedia (for general purpose concepts nodes) and a simplified version of WordNet-Affect\(^2\) (for sentiment/emotion concept nodes). Moreover, we enrich the EPG with conceptual nodes related to the TV events. In particular, we link all TV event concept nodes to a TV program node (for instance, we may link an hypothetical node concerning “Dexter, Season 4, Episode 12” to another concept node related to “Dexter, Season 4” in its turn linked to the more general concept of “Dexter (TV series)”).

VI. Querying the Knowledge Graph

The knowledge graph described in the previous section can potentially represent huge amounts of rich, heterogeneous and time-evolving information. Accessing and querying the graph in a simple but efficient way are then crucial for the usability of the system. The result-set of each query can be processed for visualization and analysis purposes. To this end, we must define a simpler model to represent the result-set of a query on the Knowledge Graph $G^K$. In particular, the result-set of a query is modeled as an undirected weighted graph $G^O = (V^O, E^O, W^O)$, where $V^O$ is a set of vertices; $E^O = \{(v_i,v_j)\. s.t. v_i,v_j \in V^O\}$ is a set of undirected edges; $W^O: V^O \times V^O \to \mathbb{R}$ is the function that associates a weight $w_{ij}$ to each edge $(v_i,v_j) \in E^O$.

As an extension, the result-set may involve multiple query graphs. Here, we consider a more general model consisting of a collection of $N$ query graphs $G^Q = \{G^Q_1, \ldots, G^Q_N\}$.

We can now define the general form of a graph query:

**Definition 5 (graph query):** Given a knowledge graph $G^K$, a query $Q^F(G^K, P, F)$ returns a collection of query graphs $G^Q = \mathcal{F}(G^K)$, where: $\mathcal{F}$ is a mapping function $\mathcal{F}_{G^K}: G^K \to G^O$ that associates vertices, edges and weights in $G^K$ to vertices, edges and weights in $G^O$; $P$ is a selection predicate on $G^K$, i.e., a function $P: G^K \to \{true, false\}$; $G^K \subseteq G^O$ is the subgraph of $G^K$ satisfying $P$.

This general definition embraces potentially any kind of selection query. However, in our system, we focus on a specific type of query called similarity query. The goal of this query is to provide a graph where two vertices are connected if they are similar enough. The weight of the edge connecting them measures the strength of their similarity. Before providing the definition of similarity query, we briefly introduce the definitions of social context of a knowledge graph node. In the following, the set of all social sources is denoted by $\mathcal{KS}$.

**Definition 6 (social context):** Given a knowledge graph $G^K = (V,E,W)$ and a node $v_i \in V$, a time interval $\Delta T = (t_{start}, t_{end})$ and a set of social sources $\mathcal{KS} \subseteq \mathcal{KS}$, the social context of $v_i$ in $\mathcal{KS}$ during $\Delta T$ is given by the undirected graph $G^O_{v_i,\Delta T,\mathcal{KS}}$ built on the subgraph of $G^K$ induced by the nodes in $V^O_{v_i,\Delta T,\mathcal{KS}} \subseteq O$ and $V^O_{v_i,\Delta T,\mathcal{KS}} \subseteq C$ where: $V^O_{v_i,\Delta T,\mathcal{KS}}$ is the set of the nodes $v_i \in O$ such that (i) $\exists (o_i,t) \in E^{wp}$ s.t. $t_{start} \leq t_{label} \leq t_{end}$, (ii) $o_i.source \in KS$, (iii) there is a path $P_3 = \{(v_i,v_{i+1}),\ldots,(v_i,v_{i+k}),\ldots, (v_i,v_{i+k+n})\}$ where $v_{i+k} = v_i$ and $\forall k=0,\ldots,n, (v_i,v_{i+k},v_{i+k+1}) \in E^{wp}$.
query that provides the similarity graph of a given set of nodes. The node similarity between two nodes is defined as follows:

Definition 7 (node similarity): Given a knowledge graph $G = (V, E, W)$, two nodes $v_i, v_j \in V$, a set of knowledge sources $KS \subseteq \mathcal{KS}$ and a time interval $\Delta T$, the similarity between $v_i$ and $v_j$, namely $\text{sim}(v_i, v_j, KS, \Delta T)$, is a function of the graph $G^c_{v_i, c_i, \Delta T} \cup G^c_{v_j, c_j, \Delta T}$ s.t. $\text{sim}(v_i, v_j, KS, \Delta T) \in \mathbb{R}$.

We can now provide the formal definition of the central notion of similarity query:

Definition 8 (similarity query): Given a knowledge graph $G^c = (V, E, W)$, a selection predicate $P$ on $V$, a mapping function $F$, a set of knowledge sources $KS \subseteq \mathcal{KS}$ and a time interval $\Delta T$, a similarity query $Q = (G^c, KS, P, F, \Delta T)$ returns the query graph $G^q_{v, v', F} = F(V')$, where $V' = \{v' \in V \mid \exists v \in V \text{ s.t. } (v, v') \in E\}$ and $E^q = \{(v, v') \in V^2 \mid v, v' \in V, v < v', \text{ and } \text{sim}(v, v', KS, \Delta T) = \text{sim}(v, v', KS, \Delta T') \neq 0\}$. $W^q \subseteq V^2 \times V^2$ is the function that associates a weight $w_{ij} = \text{sim}(v_i, v_j, KS, \Delta T)$ to each edge $(v_i, v_j) \in E^q$.

We now show how to instantiate the similarity query in the TV domain. In particular, we provide here a way to compute the entity similarity in different similarity query formulations. We are particularly interested in two types of queries: concept similarity queries and cross-source similarity queries.

A. Computing Concept Similarities
The simplest type of query is the one that involves a target subset of concept nodes $G_{target} \subseteq \mathcal{C}$. Such queries may involve nodes of the same subtype (i.e., only people) or nodes of two different subtypes (e.g., TV events and people). The subtype of a node is stored in a node attribute called subtype. Before introducing the definitions of similarity, we define the notion of path strength between two nodes:

Definition 9 (path strength): Given an undirected graph $G = (V, E, W)$ s.t. each node has a magnitude property, two nodes $v_i, v_j \in V$, and a generic path between $v_i$ and $v_j$, denoted by $p(v_i, v_j, G) = v_i \xrightarrow{c_i} v_{i+1} \xrightarrow{c_{i+1}} \ldots \xrightarrow{c_{k-1}} v_k \xrightarrow{c_k} v_j$, where $v_{i+1} = v_i$ is the strength of $p(v_i, v_j, G)$ is given by

$\text{str}(p(v_i, v_j, G)) = v_i \cdot \text{magnitude} \prod_{i=1}^{k} v_{i+1} \cdot \text{magnitude} \prod_{i=1}^{k} v_{i+1} \cdot \text{magnitude} \prod_{i=1}^{k} v_{i+1} \cdot \text{magnitude} \prod_{i=1}^{k} v_{i+1} \cdot \text{magnitude} \prod_{i=1}^{k} v_{i+1}$

Note that when the magnitude of a node or the weight of an edge are not defined, their default value is 1. Thanks to this basic definition, we may define several similarity metrics that involve any two nodes of the graphs in the following way:

Definition 10 (node similarity): Given a knowledge graph $G^c$, two nodes $v_i$ and $v_j$, a time interval $\Delta T = (t_{\text{start}}, t_{\text{end}})$ and a set of social sources $KS \subseteq \mathcal{KS}$, the similarity between $v_i$ and $v_j$ is given by

$\text{sim}(v_i, v_j, KS, \Delta T) = \sum_{p \in P} \text{str}(p(v_i, v_j, G^c_{v_i, c_i, KS, \Delta T}) \cup G^c_{v_j, c_j, KS, \Delta T}))$

where $P$ is the set of all paths in the graph resulting from the union of the social context of $v_i$ and $v_j$.

In our application, this definition is too generic, since the similarity may also involve less relevant paths. Instead, we prefer to consider a TV event-driven similarity which involves multiple social sources whose individual contribution to the metric can be controlled by the user. Furthermore, we focus on the distance between the subset of concepts corresponding to the named entities. To this purpose we employ the similarity between a named entity node $e_i$ and a TV event node $v^T_j$ by considering only a social source $ks \in KS$. This similarity is noted $\text{sim}(v^T_j, e_i, \{ks\}, \Delta T)$. Using this similarity function, we can compute the so-called TV event-based named entity similarity between any two named entity concept nodes. In detail, given a set of TV events $C^{TV} \subseteq \mathcal{C}$, we associate, to each named entity concept $e_i \in C$, a vector $e_i = \{e_i, \ldots, e_h, \ldots, e_n\}$ where $n = |C^{TV}|$ and $e_h = \text{sim}(e_i, e^{TV}_j, \{ks\}, \Delta T)$. The TV event-based similarity between two concepts $e_i$ and $e_j$ is then given by:

$\text{sim}_{TV}(e_i, e_j, \{ks\}, \Delta T) = \frac{1}{1 + \sum_{k=1}^{n} |e_h - e_j|^2}$

i.e., the similarity between two concepts is inversely proportional to the Euclidean distance between their corresponding TV event vectors. Notice that $\text{sim}_{TV}(e_i, e_j, \{ks\}, \Delta T) \in (0, 1]$. So far, we have considered each source as equivalent. However we may assign a different weight to each knowledge source in order to let the user control the importance of each source in computing the similarity. To this purpose, we slightly modify the definition of TV event-based named entity similarity by introducing an importance coefficient $a_k$ for each source $ks \in KS$ (note that $\sum_k a_k = 1$). The weighted TV event-based named entity similarity is then given by

$\text{sim}_{TV}(e_i, e_j, KS, \Delta T) = \sum_{k \in KS} a_k \cdot \text{sim}_{TV}(e_i, e_j, \{ks\}, \Delta T)$.

B. Named Entity Similarity Graph
In Section VI-A, we have provided the application-specific notion of TV event-based similarity. We now have all the necessary components to better describe how a named-entity similarity graph looks like. Let $V^{TV} \subseteq \mathcal{C}$ be a set of $m$ named entity concepts and $V^{TV} \subseteq \mathcal{C}$ a set of TV events nodes. The
In the model, concepts represent the connecting elements among the social objects extracted from multiple heterogeneous sources.
The analysis of an individual source, without our knowledge integration framework, would have led to a different, less precise, social network.

### A. Social Centrality Study

The first example we consider here concerns the study of the importance (in terms of centrality) of persons (politicians, television people, presenters, guests), during the observation period. To perform this analysis, we consider all the persons referred by the tweets and Facebook posts associated to all episodes of the TV show and build the underlying social network following the definition of *TV event-based named entity similarity graph* (see Definition 11 in Section VI-B). According to this definition, there is an edge between two person nodes if there exists a path between these two persons, traversing a *TV Event* node. Interestingly, these paths may involve cross-source nodes, i.e., the analysis of an individual source, without our knowledge integration framework, would have led to a different, less precise, social network. Since more than one tweetset and Facebook post may exist during the week associated to each episode, for each episode, the similarity graph is such that all the tweetsets and Facebook posts are merged to obtain an aggregated episode representation. Moreover, each of them is associated to the set of the most mentioned persons during the considered period.

On our Twitter data, the above described analysis produced a social network that we analyzed by computing the betweenness centrality [34], [35] of each node (i.e., the number of shortest paths from all vertices to all others that pass through that node), obtaining the results in Table 2 (a). These results show that Matteo Renzi is very central for this TV program. He is the Italian prime minister and he was strongly involved in each reform cited before. It is important to note that he is the most central concept in terms of betweenness even if he participated only in few episodes during the observation period. The following concept is Matteo Salvini, that is one of the most active leaders of the center-right coalition especially known for his stance against illegal immigration and his strong criticism on the rules coming from the European Commission about economy and immigration.

The betweenness centrality computed for the Facebook data on people belonging to the knowledge graph is reported in Table 2 (b). The most important concept in terms of centrality is always Matteo Renzi, but, interestingly, his centrality appears less dominant w.r.t. other people. This analysis shows that one of the best ranked persons is Beppe Grillo. This is probably due to the fact that Grillo’s supporters are particularly active in this social media platform. Thus, in this social network, the position of Grillo is more central than in the Twitter case.

By combining the two information sources, we may notice that all the relevant information for both sources is preserved, as shown by the ranked betweenness scores in Table 2 (c). In particular, Matteo Renzi, the Prime Minister is still the most central concept and, almost all the most important Italian politicians actors are in the first positions of the ranking.

### B. Popularity Study

The second experiment consists in computing the “episode popularity” of each person. The popularity of a given node is related to the percentage of citations of the associated persons’ names in tweets and Facebook post comments. Notice that this information is stored in the knowledge graph as the weight of the edge connecting each person to the *People* node, by the resource extractors. Hence, to conduct this analysis, we only need to aggregate the weights of the out-edges of each person node. Within a single source the aggregation is performed by merging all social objects (a tweetset or a Facebook post) related to a given episode. Then, each edge weight is multiplied by the total number of occurrences of the concept node *People*. Finally, the cut-off method based on energy is employed to filter out the less important entries. To consider the popularity in both Twitter and Facebook as a whole, we merged the Facebook post nodes and Tweetset nodes associated to each episode. The resulting weight for each person node $i$ is then computed as

$$w(i) = \alpha \cdot w(i)_T + (1 - \alpha) \cdot w(i)_F,$$

where $w(i)_T$, $w(i)_F$ and $w(i)$ are, respectively, the node weights of the edge connecting $i$ to the Tweetset node, the node weights of the edge connecting $i$ to the Facebook post node, and the resulting weight associated to the edge connecting $i$ to the aggregated social object node. In this experiment, we considered all sources with the same weight, i.e., $\alpha = 0.5$.

| TABLE 2 Top betweenness centrality scores of nodes from different social networks. |
|---------------------------------|-----------------|
|                                 | PERSON          | CENTRALITY     |
| (a) TWITTER SOCIAL NETWORK      |                 |                |
| 1                               | MATTEO RENZI    | 0.5244         |
| 2                               | MATTEO SALVINI  | 0.1624         |
| 3                               | SILVIO BERLUSCONI | 0.0477      |
| 4                               | MURIZIO LANDINI | 0.0379         |
| 5                               | ELSA FORNERO    | 0.0139         |
| (b) FACEBOOK SOCIAL NETWORK     |                 |                |
| 1                               | MATTEO RENZI    | 0.1662         |
| 2                               | MASSIMO GIANNINI | 0.1550       |
| 3                               | SILVIO BERLUSCONI | 0.1159     |
| 4                               | MATTEO SALVINI  | 0.0714         |
| 5                               | BEPPE GRILLO    | 0.0655         |
| (c) COMBINED SOCIAL NETWORK     |                 |                |
| 1                               | MATTEO RENZI    | 0.3089         |
| 2                               | MATTEO SALVINI  | 0.1208         |
| 3                               | SILVIO BERLUSCONI | 0.1008     |
| 4                               | MASSIMO GIANNINI | 0.0689      |
| 5                               | MURIZIO LANDINI | 0.0362         |
Figure 2 shows the results for the top-ranked personalities. As can be seen, the popularity of Matteo Renzi is quite stable during the observation period, while the popularity of Maurizio Landini, one of the most important trade unionists, is strongly related to episodes in which the main theme was the labor reform.

C. An Example of Cross-Source Analysis
As an example of the potential analysis scenarios that our framework may enable, we consider non-trivial associations between Facebook posts and Twitter hashtags. These two objects are not immediately linked; users’ communities and social platforms are different. However, they may have in common several entities (persons, nouns, events, and emotions). Thanks to our framework, it is rather simple to compute the entities that connect posts and hashtags. We then construct a hashtags × post matrix (called $M$) in the following way. For a given post $p$ and a given hashtag $h$, the value $m_{hp}$ of matrix $M$, is given by $m_{hp} = \text{sim}_{\text{co}}(o_h, o_p, \Delta t)$, where $o_h$ is the social object associated to the hashtag $h$, $o_p$ is the social object associated to the Facebook post $p$, $\Delta t$ is the whole six months analysis period and $\text{sim}_{\text{co}}$ follows Definition 12 given in Section VI-C. We repeat this computation for each pair $(h, p)$ of hashtags $h$ and posts $p$. We ignore all concept nodes related to emotions in this case. As a result, the association of all posts and tweets related to the monitored period, leads to a matrix $M$ of 179 most used hashtags and 1,679 Facebook posts, consisting of 144,023 non-zero values.

It is now interesting to obtain associations between groups of hashtags and groups of Facebook posts. As an example, we may imagine cross-domain recommendation of interesting Twitter hashtags to people reading and commenting Facebook posts. To compute relevant cross-associations, we use a hierarchical co-clustering algorithm [36]. It identifies a hierarchy of clusters of rows and an associated hierarchy of clusters of columns by optimizing the Goodman-Kruskal’s $\tau$ association measure. The algorithm is parameter-less, and build compact hierarchies with n-ary splits. We apply this algorithm and consider two coupled levels of the hierarchy; the first level, with a coarse grid of $3 \times 3$ co-clusters; the third level with a more fine-grained grid of $14 \times 20$ co-clusters.

We then associate each cluster $R$ of rows (Facebook posts) with the cluster $C$ of columns (hashtags) such that $\max_{\sum_{(h, p) \in C} m_{hp}}$ is maximized.

**FIGURE 2** Episode popularity (computed according to Equation 1) of some cited persons from our knowledge graph.

**FIGURE 3** Two examples of hashtag clusters about (a) economic reforms and (b) foreign politics.
As an example, we considered two results at different level of the hierarchical co-clustering. In the former, considering the first hierarchical level, one of the row co-cluster contains 164 Facebook posts that are associated to a cluster of 72 hashtags. As can be seen in Fig. 3a), they are mostly related to discussions about the highly debated economic reforms and school. The second example (see Fig. 3b)) is extracted from the third level of the hierarchical clustering and associates a set of 49 Facebook posts to the reported set of hashtags: as can be seen, the set of terms depict the discussion, mostly debated by the center-right parties, around immigration and terrorism.

VIII. Conclusions and Future Work

In this paper, we proposed an integration framework in which TV users’ activities on different social media are collectively represented, and possibly enriched with external knowledge, such as information extracted from the EPGs, or available ontological domain knowledge. We also discussed different types of analysis that the integration data model enables.

Many research problems remain open. As future work, we will be able to scalar issues which immediately emerge when engineering an industrial system based on the presented framework. The intense activity of social media users turns into the high dynamicity of the knowledge graph. Hence, we are studying incremental (possibly approximate) versions of the algorithms computing graph based popularity measures.

The data model allows us to track the temporal evolution of users’ activities. Thus, another future work includes the application of innovative algorithms and techniques to analyze time series extracted from the graph. This will allow us to capture and study how social phenomena (popularities, users’ interests, and communities of users sharing common interests) evolve in time.

Finally, for future work, we plan to leverage the most recent research results in social media analytics and sentiment analysis to further improve our framework. In particular, we will adopt some event detection techniques (such as the one presented in [37]) to support the automatic detection of emerging topics in social media and we will consider sentiment computing [38], [39] and AffectiveSpace [40], to bring sentiment analysis up to concept-level.

Acknowledgments

We are grateful to Alessio Antonini, Roberto Del Pero and Fulvio Negro for their constructive discussions during the formalization of the integration framework.

References

An Efficient Memetic Algorithm for Influence Maximization in Social Networks

I. Introduction

In recent years, online social network sites such as Facebook and Twitter have enjoyed an increasing attention. They are becoming popular platforms selected by companies for promoting their products and spreading information with the number of users growing rapidly. In these websites, users are enthusiastic about interacting, sharing and collaborating through online social media which makes information spread among users more easily [1]–[4]. The word-of-mouth marketing or viral marketing, an effective strategy on social network marketing, aims to produce a maximal cascading effect in social networks by targeting only a small number of selected users. Specifically, to promote a new product in a social network, the pre-existing adopters will make a recommendation to their friends, and then their friends will make a recommendation to their friends' friends and so on. This strategy is proved to be cost-effective and successful [5]–[6]. However, the crucial problem is how to select...

Abstract—Influence maximization is to extract a small set of nodes from a social network which influences the propagation maximally under a cascade model. In this paper, we propose a memetic algorithm for community-based influence maximization in social networks. The proposed memetic algorithm optimizes the 2-hop influence spread to find the most influential nodes. Problem-specific population initialization and similarity-based local search are designed to accelerate the convergence of the algorithm. Experiments on three real-world datasets demonstrate that our algorithm has competitive performances to the comparing algorithms in terms of effectiveness and efficiency. For example, on a real-world network of 15233 nodes and 58891 edges, the influence spread of the proposed algorithm is 12.5%, 13.2% and 173.5% higher than the three comparing algorithms Degree, PageRank and Random, respectively.

Maoguo Gong, Chao Song, Chao Duan, Lijia Ma, and Bo Shen
Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education, Xidian University, Xi’an, CHINA

Corresponding Author: Maoguo Gong (E-mail: gong@ieee.org).
Influence maximization is to extract a small set of nodes (or seeds) from a social network which can generate the propagation maximally under a cascade model.

The crucial problem above, known as influence maximization, is to extract a small set of nodes (or seeds) from a social network which can generate the propagation maximally under a cascade model [1], [7]–[14]. This problem is first studied by Domingos and Richardson [6]. They view a market as a social network and propose a probabilistic method with modeling the influence between network users as a Markov random field. Then the problem is formalized as a discrete optimization problem which is proved to be NP-hard under the independent cascade model (IC) and the linear threshold model (LT) by Kempe et al. [15]. A natural hill-climbing greedy strategy is also proposed by them as an approximate algorithm to solve the optimization problem. However, the greedy algorithm has two intrinsic difficulties to be applied to large social networks [7]. Firstly, the greedy algorithm needs to traverse all remaining nodes in the network when it selects each next seed [7]. As a result, the computation of the greedy algorithm has a quadratic relationship with the number of nodes [7]. Secondly, computing exact influence spread of a node set under the IC model and the LT model is \#P-hard [7]. Therefore, a Monte-Carlo simulation is run to obtain an accurate estimation. However, this needs a large number of runs, typically 10,000 times, which results in a large computation time.

Memetic Algorithms (MAs) [16], a branch of evolutionary computation, are hybrids of global search methods and local search procedures [17]–[21]. The global search methods are generally evolutionary and swarm intelligence [17]. They can produce a reliable estimate of the global optimum. The local search procedures are individual refinement processes incorporating domain-knowledge. They can explore better solutions around the best solution found so far [22]–[23]. MAs have been proved to play an important role in solving complex optimization problems in social networks.

In [24], a memetic algorithm named as Meme-Net is proposed to solve the resolution limit problem in community detection. In [25], a fast memetic algorithm is presented to solve community detection effectively. The algorithm adopts a multi-level learning strategy as the local search procedure and...
shows its effectiveness. In [26], a fast memetic algorithm is proposed to compute and transform structural balance in signed networks.

In this paper, we propose a novel memetic algorithm for influence maximization in social networks, termed as CMA-IM. Fig. 1 provides the framework of CMA-IM, which comprises three steps: (1) Network clustering; (2) Candidate selection; (3) Seed generation. In the first step, we use a fast two-phase heuristic algorithm BGLL [27] to detect communities in networks. In the second step, we first select the communities that are significant and then choose a few nodes from each significant community to form the candidate pool. In the final step, we model the influence maximization problem as the optimization of a 2-hop influence spread [13] and propose a problem-specific memetic algorithm to find the ultimate seeds. The proposed algorithm combines a genetic algorithm as the global search method and a similarity-based local search procedure, which can accelerate the convergence of the algorithm. In order to solve the problem of influence overlapping, we also apply the similarity-based strategy to the processes of crossover and mutation.

We estimate the effectiveness and the efficiency of CMA-IM on real-world networks. The experimental results show that our algorithm has competitive performances to the comparing algorithms.

The rest of this paper is organized as follows: Section II reviews related works. Section III gives a description of the problem model. Section IV presents our proposed CMA-IM algorithm in detail. Section V shows experimental performances of the proposed algorithm on three real-world networks. Finally, the conclusion is summarized in Section VI.

II. Related Works

Many methods have been proposed for influence maximization [1], [7], [9]–[12]. Leskovec et al. [9] present a method called CELF which is reported to be 700 times faster than the greedy algorithm. They develop a lazy procedure when selecting new seeds. The lazy procedure exploits the submodularity of the spread function to reduce the number of function evaluations. Goyal et al. [7] propose an improvement on [9] named as CELF++ which further improves the efficiency. Chen et al. [10] put forward two new greedy algorithms. One is the New-Greedy. It attempts to reduce computations by generating a new smaller graph with all edges not participating in the propagation removed. The other one is the MixedGreedy. It runs the New-Greedy in the first round and runs the CELF [9] in the later rounds. Although these improvements outperform the greedy algorithm in efficiency, they still cannot be scalable to large-scale networks. Recently, some authors use the communities of the networks to improve the efficiency of algorithms. Wang et al. [11] introduce a community-based greedy algorithm called CGA which narrows down the search space of the influential nodes from the whole graph to the communities. They exploit dynamic programming to select the community which has the largest increase of influence spread and adopt MixGreedy algorithm [10] to find the most influential node as the seed in the chosen community. Chen et al. [1] propose a community-based algorithm under the Heat Diffusion Model, termed as CIM. They select seeds from the candidate nodes by comparing scores of them. Rahimkhani et al. [12] present a fast algorithm, called ComPath, combining with the community character and introduce an influence spread estimation under the LT model.

In this paper, network clustering is completed by BGLL which is different from the network clustering algorithms used in [1] and [12]. In [1], the authors propose a hierarchical clustering (HClustering) method which iteratively merges nodes into communities based on the structural similarity between each pair of nodes. In [12], the authors employ the SLPA algorithm [28] which discovers overlapping communities according to the listener-speaker interaction rules.

We propose a problem-specific population initialization method and a similarity-based local search procedure, which can accelerate the convergence of the algorithm. To
III. Problem Model

We model a social network as an undirected network denoted as \( G = (V, E) \) where \( V \) represents the node set denoting users in the social network and \( E \) represents the edge set denoting the relationships between users [10]. \( N \) and \( M \) are the number of nodes and edges, respectively.

Given a node set \( S \) that includes \( k \) nodes, the influence spread produced by \( S \) which is denoted as \( \sigma(S) \) is the number of nodes that \( S \) can influence in the network \( G \) under a certain cascade model. In this paper, we select the IC model as the cascade model, which is widely used in the previous works [7], [9], [10], [13], [15]. In the IC model, the state of a node has only two types, either active or inactive. The inactive nodes can be changed into the active nodes, but not vice versa. Each edge is associated with a propagation probability \( p \) and \( p(u, v) \) represents the probability of an inactive node \( v \) to be influenced by its active neighbor \( u \). Under the IC model, \( \sigma(S) \) works as follows. Let \( S \) be the set of nodes that are active in the step \( t \), and \( S_0 = S \) which is initialized by a \( k \)-node set. At step \( t \), each node \( v \in S_{t-1} \) has only one chance to independently activate every inactive neighbors with the probability \( p \). This influence diffusion process stops at the step \( t \) when \( S_t = \emptyset \) and \( \sigma(S) \) is the union of \( S_t \) obtained at each step. Thus, the influence maximization problem is to find the \( k \)-node set \( S \) that can make \( \sigma(S) \) maximal under the IC model. Let us illustrate this problem using the greedy algorithm as an example on a toy network in Fig. 2.

Suppose that we attempt to find a 2-node set \( S \) that can influence the most number of nodes of the network in Fig. 2. When it is to find the first seed, the greedy algorithm traverses all the nodes from 1 to 10 and calculates the influence spread of each node that is estimated under the IC model. It selects node 6 which has the maximum influence spread as the first seed. To find the second seed, the greedy algorithm computes the influence spread of \((6, 1), (6, 2), \ldots, (6, 5), (6, 7), \ldots, (6, 10)\) and eventually it selects node 3 as the second seed, because it results in the most increase of the influence spread. The process to find the 2-node set \( S = \{6, 3\} \) which possesses the maximal influence is the influence maximization problem. Here, the influence spread of a node set is calculated by running 10,000 times a Monte-Carlo simulation.

As mentioned above, computing exact \( \sigma(S) \) under the IC model is \#P-hard and it needs to run a number of Monte-Carlo simulations. And with the size of a social network growing larger, the time consumed by running Monte-Carlo simulation becomes not negligible.

In [13], Lee et al. propose a fast approximation for influence maximization, they consider the influence spread on the nodes within 2-hops away from seed node set instead of all nodes in the network. The 2-hop influence spread of a node set \( \hat{\sigma}(S) \) is calculated as (1).

\[
\hat{\sigma}(S) = \sum_{i \in S} \sigma_i \left( \sum_{j \in S} \sum_{c : j \in N_C} p(s, j)(\sigma'_i - p(s, j)) \right) - x.
\]

where \( C \) denotes the 1-hop nodes cover of node \( s \), i.e., the neighbors of node \( s, p \) is the propagation probability in the IC model, \( x = \sum_{i \in S} \sum_{c \in N_i} \sum_{j \in N_C} p(s, j)p(c, d) \) and \( \sigma'_i \) is the 1-hop influence spread of node \( c \). In (1), the first term is the sum of the 2-hop influence spread of each seed in \( S \), the second term considers the redundant situation that a seed is a neighbor of another seed and the third term considers the redundant situation that a seed is 2-hops away from another seed.

The authors show that the 2-hop influence spread is sufficiently valid and efficient to estimate the influence spread of a node set. Therefore, we adopt the 2-hop influence spread in our algorithm. The important variables mentioned above are listed in Table 1.

IV. The Proposed Algorithm for Influence Maximization

In this section, we will introduce our proposed algorithm CMA-IM. As illustrated earlier in Fig. 1, CMA-IM consists of three steps: (1) Network clustering; (2) Candidate selection; (3) Seed generation. A detailed description of each step will be given in the following.

![Figure 2](image-url)  
**Figure 2** An illustration for the influence maximization problem using the greedy algorithm on a toy network. Node 6 is the first selected seed and node 3 is the second selected node.

<table>
<thead>
<tr>
<th>TABLE 1: The variables used in this paper.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARIABLES</td>
</tr>
<tr>
<td>( G(V, E) )</td>
</tr>
<tr>
<td>( N )</td>
</tr>
<tr>
<td>( M )</td>
</tr>
<tr>
<td>( k )</td>
</tr>
<tr>
<td>( S )</td>
</tr>
<tr>
<td>( p )</td>
</tr>
</tbody>
</table>
Social networks naturally tend to be clustered into groups or communities. The nodes connect more densely with the nodes in the same group than the nodes outside the group.

A. Network Clustering

Social networks naturally tend to be clustered into groups or communities [29]. The nodes connect more densely with the nodes in the same group than the nodes outside the group. Clustering into communities is a property of social networks, which is beneficial for understanding the structure of networks [30]. Modularity, a famous criterion proposed by Newman and Girvan [29], provides a quality evaluation of the network community structure. Various algorithms for network clustering are based on modularity optimization including heuristic algorithms [27], [31]–[33] and evolutionary algorithms [24], [30].

BGLL proposed by Blondel et al. [27] is a fast heuristic method based on modularity optimization, which consists of two phases. At the first phase, each node of the network is considered as a community. Then they remove a node from its original community to its neighbor’s community which has the maximal positive gain in modularity. This phase is applied repeatedly for all nodes until no further improvement can be achieved. The first phase is then complete [27]. The second phase considers the communities obtained in the first phase as nodes such that a new network can be built. Then BGLL runs these two phases iteratively until achieving an unchanged result and obtaining the maximal modularity.

Compared with the HClusreing and SLPA algorithm, the BGLL algorithm can discover more natural structures of networks because it needs no prior knowledge about the community number. So communities obtained by the BGLL get closer to the inherent communities in networks. Meanwhile BGLL only needs a few iterations to obtain a maximal modularity, which makes the BGLL algorithm have better performance in terms of efficiency when applied to large-scale networks.

B. Candidate Selection

The candidate selection step aims to determine a set of candidate nodes according to the information about communities obtained in the first step. Because social networks in realistic settings are usually extremely huge, the search space for selecting seeds is also huge. Therefore, there is a need to effectively reduce the number of candidate nodes.

By analyzing the structures of communities, we find that not all communities are significant enough to accommodate seed nodes. For example, in Fig. 3, although the network is divided into three communities, community 3 may be insignificant compared with community 1 and 2 due to its smaller community size. Suppose we choose a seed node from community 3, it may only activate three nodes initially. We choose the community 1 and 2 as the significant communities because their sizes are large and there may be more influential nodes in them. Here, we define significant communities as the first n large communities, where n is varied with networks. However, the nodes in community 1 have many common neighbors. For example, node 3 and node 6 have 5 common neighbor nodes, which is not beneficial to influence spread. In order to solve the problem of overlapping influence, we propose a similarity-based high degree method called SHD which is described in Section IV-C1.

Let Candidate be a candidate node pool. Next, our task is to choose a number of potential nodes from each significant community to fill in the Candidate. If considering the influence ability of a node, degree centrality may be the most intuitive way to estimate the ability. The higher degree of a node is, the more neighbors the node has, which means that node with higher degree can influence more nodes with the same propagation probability. Therefore, we select potential nodes from each significant community based on the degree centrality. Here, we choose the way of [12], which is shown as (2), to decide the number of candidate nodes selected from each significant community.

\[
\text{Candidate size} = \left( \frac{C_i - \text{MinC}}{\text{MaxC} - \text{MinC}} \right) \times \beta + \alpha,
\]

where \(C_i\) is the size of the i-th significant community, MaxC is the size of the largest significant community and MinC is the size of the smallest significant community. The term \((C_i - \text{MinC})/(\text{MaxC} - \text{MinC})\) is the ratio of the i-th community among all selected communities and its value is confined to \([0, 1]\). \(\beta\) is the amplification term and \(\alpha\) is the constant.
term that guarantees the least selection in each community. After selecting the *Candidate*, the last step of our algorithm is to generate the ultimate seeds.

**C. Seed Generation**

After the two aforementioned steps, the search space has been reduced. In the next step, we will employ the proposed problem-specific memetic algorithm, named as Meme-IM, to generate the ultimate seeds by optimizing the 2-hop influence spread shown as (1). The whole framework of Meme-IM is shown as Algorithm 1.

In Step 1), Meme-IM mainly completes the population initialization task. Firstly, it creates the initial population of solutions \( P = \{x_1, x_2, ..., x_{\text{pop}}\} \)\(^T\) according to a problem-specific strategy. And then it selects the individual with the maximum fitness as \( P_{\text{best}} \). Step 3) is the evolution procedure. In Step 3.1), Meme-IM first uses the deterministic tournament selection method to select parental individuals \( P_{\text{parent}} \) for mating in genetic algorithm. Then in Step 3.2), Meme-IM reproduces the chosen parental individuals \( P_{\text{parent}} \), i.e., performs crossover and mutation operation on \( P_{\text{parent}} \). Step 3.3) is an individual reinforcement procedure. Step 3.4) is to refresh the current population by taking the best \( \text{pop} \) individuals from \( PU P_{\text{best}} \). And in Step 4), when the algorithm terminates on convergence, Meme-IM stops and outputs the ultimate \( k \)-node set.

**Algorithm 1 Framework of Meme-IM.**

**Input:** Maximum generation: \( \text{maxgen} \), population size: \( \text{pop} \), mating pool size: \( \text{pool} \), tournament size: \( \text{tour} \), crossover probability: \( \text{pc} \), mutation probability: \( \text{pm} \), spread probability: \( \rho \), seed size: \( k \), the candidate nodes pool: \( \text{Candidate} \) and the connection matrix: \( A \).

**Output:** The most influential \( k \)-node set.

1: **Step 1** Initialization
2: 1.1) Population initialization:
   \[ P = \{x_1, x_2, ..., x_{\text{pop}}\}^T \]
3: 1.2) Best individual initialization: \( P_{\text{best}} = x_i \)
4: 2) Set \( t = 0 \); // the number of generations
5: 3) Repeat
6: 3.1) Select parental chromosomes for mating:
7: \( P_{\text{parent}} \leftarrow \text{Selection}(P, \text{pool, tour}) \)
8: 3.2) Perform genetic operators:
9: \( P_{\text{next}} \leftarrow \text{GeneticOperation}(P_{\text{parent}}, \text{pc, pm}) \)
10: 3.3) Perform local search:
11: \( P_{\text{next}} \leftarrow \text{LocalSearch}(P_{\text{next}}) \)
12: 3.4) Update population:
13: \( P = \text{UpdatePopulation}(P, P_{\text{next}}) \)
14: 3.5) Update the best individual \( P_{\text{best}} \)
15: 4) Stopping criterion: If \( t < \text{maxgen} \), then \( t = t + 1 \) and go to Step 3), otherwise, stop the algorithm and output.

In the following, we will give a more detail description of several important procedures including the population initialization, genetic operation and local search procedure.

1) **Representation and Initialization**

In Meme-IM, each chromosome (individual) \( x_i \) \((1 \leq i \leq \text{pop})\) in the population represents a \( k \)-node set, which is encoded as an integer string

\[ x_i = \{x_i^1, x_i^2, ..., x_i^k\} \]

where \( k \) is the number of seeds, each gene \( x_i^j \) of the chromosome corresponds to a node selected from \( \text{Candidate} \). An illustration of this representation is shown as Fig. 4. It is noticed that there is no repeated node in \( x_i \). Considering that the solution produced by selecting \( k \) nodes randomly from \( \text{Candidate} \) is of low quality and may result in a long time to converge, we attempt to initialize a higher quality population to speed up the convergence. Here, we propose a similarity-based high degree method called SHD. The SHD and the random mechanism can guarantee the convergence and diversity of the individuals. The population initialization procedure is shown as Algorithm 2.

High degree centrality is a standard method for influence maximization on social and other networks [34]. But high

**Algorithm 2 Population initialization.**

**Input:** Population size: \( \text{pop} \)

**Output:** Population \( P \)

1: Generate a half of population based on SHD, see Algorithm 3 for more information;
2: for \( i \) from 1 to \((\text{pop}/2)\) do
3: for \( j \) from 1 to \( \text{pop} \) do
4: if \( \text{rand}(1) > 0.5 \) then
5: select a random node different from each node in \( x_i \) from the \( \text{Candidate} \) to replace \( x_i^j \)
6: end if
7: end for
8: end for
9: for \( i \) from \((\text{pop}/2 + 1)\) to \( \text{pop} \) do
10: select \( k \) different nodes from the \( \text{Candidate} \) to initialize \( x_i \) based on SHD
11: end for

**FIGURE 4** Illustration of the representation. Left: a 9-node set selected from \( \text{Candidate} \). Right: the individual encoding of the 9-node set.
degree centrality may produce overlapping influence spread between nodes. To solve this problem, we propose a similarity-based high degree method (SHD). SHD starts with choosing the node with the highest degree in Candidate. After choosing a node, it excludes the neighbor nodes that are similar with the existing nodes. Then SHD chooses the next node with the highest degree in the left candidate nodes. This procedure iteratively operates until \( k \) nodes are chosen. Here, the degree of similarity between nodes is measured by the structure similarity defined as (3). The process is described as Algorithm 3.

In Algorithm 3, \( N(v) = \{ u \in V \mid uv \in E \} \) represents the neighbors of node \( v \) and the structural similarity between nodes \( u \) and \( v \) is defined as (3).

\[
\text{Similarity}(u,v) = \frac{|NB(u) \cap NB(v)|}{|NB(u)| + |NB(v)|}
\]  

where \( NB(v) = \{ u \mid v \cup N(v) \} \) includes the node \( v \) and its neighbors. In Algorithm 3, \( \text{sim} \) is a threshold confined to \([0,1]\) which is set according to different datasets. When the structure similarity between two nodes is larger than \( \text{sim} \), the nodes are similar. The similarity between nodes is a significant criterion to prevent from overlapping influence spread.

### 2) Genetic Operators

**Crossover**

In Meme-IM, we employ the one-point crossover because it is simple. The one-point crossover works as follows. Given two parent chromosomes \( x_a \) and \( x_b \), we first randomly select a crossing over position \( i \) (\( 1 \leq i \leq k \)), then exchange each node \( j \) after the position \( i \) between the two parents \((i.e., x'_a \rightarrow x_b, \forall j \in \{ j \mid i \leq j \leq k \}\)) and then two new offspring chromosomes \( x'_a \) and \( x'_b \) return. We also should guarantee the validity of \( x'_a \) and \( x'_b \), i.e., there are no same nodes in \( x'_a \) and \( x'_b \), respectively. Specifically, when the \( j \)-th node in \( x_b \) (\( x_a \)) is not similar with the nodes in \( x_b \) (\( x_a \)) except for \( x'_b \) (\( x'_a \)), then we exchange \( x'_b \) and \( x'_a \).

**Mutation**

Here, we employ the similarity-based mutation on the generated population after crossover. For each gene in a chromosome, if the generated random value \( r \in [0,1] \) is smaller than the mutation probability \( pm \), then we mutate the gene to another gene in Candidate. The other gene is selected randomly from the genes which are dissimilar with the genes in the chromosome. The similarity is evaluated by (3). However, when the value of \( r \) is larger than \( pm \), there is also a situation we mutate the gene. When the gene is similar with the most influential gene in the chromosome, we mutate the gene to another gene in Candidate which is dissimilar with the most influential gene and its similar neighbors.

### 3) Local Search Procedure

Local search procedure is an individual reinforcement procedure which is to find a better solution around the best solutions found so far [23]. For a chromosome, when we change a node to another node in the Candidate which is different from any node in the chromosome, a neighbor of the chromosome is obtained. Here, we employ a similarity-based strategy as the local search procedure. The procedure is performed on an arbitrary individual in the population, then it attempts to find a better individual from the neighborhoods of the individual. The fitness is calculated by (1) and the neighbor individual is better if its fitness is larger than that of the original individual. If the change can produce a better individual, this change is accepted. The procedure repeats until no further improvement can be made. Here, we find the fittest chromosome in \( P_{\text{best}} \) which is obtained after the genetic operators and apply the local search procedure on it. The implementation of the local search procedure is shown as Algorithm 4.

In Algorithm 4, we first perform FindBest() function to select the individual with the maximum fitness in the input individuals. Then we apply the local search procedure on it.

![Algorithm 3 SHD algorithm.](image)

![Algorithm 4 The local search procedure.](image)
V. Experimental Study

In this section, we evaluate the effectiveness and the efficiency of our proposed algorithm CMA-IM and compare the influence spread and the running time with other six algorithms on three real-world social networks.

A. Experiment Setting

1) Datasets

Dolphin network [35]. The small size Dolphin social network describes the associations between 62 bottlenose dolphins living in Doubtful Sound, New Zealand [35]. Lusseau observed the behavior of the dolphins in a period of seven years. The nodes of the network represent dolphins and edges represent a statistically frequent associations between these dolphins.

NetGRQC network [36]. The medium size NetGRQC network is a collaboration network whose nodes represent authors and edges represent co-authors relationships between them. If two authors coauthor a paper, an edge establishes between them. The data contains papers collected from the “General Relativity and Quantum Cosmology” section of the e-print arXiv (http://www.arXiv.org) in a period from January 1993 to April 2003.

NetHEPT network [10]. The large size NetHEPT network is also a collaboration network of paper co-authors and is frequently used in previous works such as [10]. The papers in this dataset are obtained from the “High Energy Physics-Theory” section of the e-print arXiv from year 1991 to year 2003. This network contains multiedges if the two co-authors have collaborated multiple papers.

The basic characteristics of the real-world networks described above are given in Table 2.

2) Comparing Algorithms

CGA-IM and MA-IM. CGA-IM is the variant version of CMA-IM by removing the local search procedure and MA-IM is the variant of CMA-IM by removing the network clustering step. We compare CMA-IM with CGA-IM and MA-IM on two real-world networks to demonstrate the effectiveness of the network clustering and the local search procedure in CMA-IM.

CELF. The CELF algorithm [9] is an improvement on the greedy algorithm which has the same result as the greedy algorithm and as much as 700 times speedup. Here, we take the number of Monte-Carlo simulations as 10,000 to obtain an accurate estimate.

CMA-HClustering and CMA-SLPA. CMA-HClustering and CMA-SLPA are two comparing algorithms combining our memetic algorithm and the network clustering algorithms in [1] and [12], respectively.

The similarity-based local search procedure attempts to find a better individual from the neighborhoods of the individual.

Degree centrality. The Degree centrality [15] is a classical heuristic for mining influential nodes, which believes the more neighbor nodes connected, the more important the node is. It sorts nodes in the descending order according to their degrees and simply selects the top k nodes as the seeds.

PageRank. PageRank is the popular algorithm for ranking webpages based on their importance proposed by Brin and Page [37]. Here, the restart parameter is set as 0.15 and the stop criterion is set as 0.001. PageRank sorts the nodes according to their importance and return their ranks. We select the top k nodes as seed nodes.

Random. Random is a baseline method used in [15]. The random method randomly selects k nodes as the seeds.

Other famous heuristics are not taken into consideration such as the distance centrality and the betweenness centrality because of their high computational cost [10].

All experiments are implemented under the IC model. In order to compare the accuracy of different algorithms, we compute the influence spread of the ultimate k-node set of each algorithm by running Monte-Carlo simulation for 10,000 times and take the average influence spread. All algorithms are independently run 30 times on each network. All the experiments are conducted on a PC with 1.70 GHz Inter Core i5 and 8.00 GB Memory. The experimental parameters of our algorithm are listed in Table 3.

TABLE 2 Statistics of the three real-world networks.

<table>
<thead>
<tr>
<th>NETWORK</th>
<th>NODES</th>
<th>EDGES</th>
<th>AVERAGE DEGREE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOLPHIN</td>
<td>62</td>
<td>159</td>
<td>5.129</td>
</tr>
<tr>
<td>NETGRQC</td>
<td>5242</td>
<td>14496</td>
<td>5.5261</td>
</tr>
<tr>
<td>NETHEPT</td>
<td>15233</td>
<td>58891</td>
<td>3.8635</td>
</tr>
</tbody>
</table>

TABLE 3 The parameters in our algorithm.

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>MEANING</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>α</td>
<td>The constant term in (2)</td>
<td>4</td>
</tr>
<tr>
<td>β</td>
<td>The amplification term in (2)</td>
<td>10</td>
</tr>
<tr>
<td>maxgen</td>
<td>The maximum generation</td>
<td>50</td>
</tr>
<tr>
<td>pop</td>
<td>Population size</td>
<td>200</td>
</tr>
<tr>
<td>pool</td>
<td>Size of the mating pool</td>
<td>100</td>
</tr>
<tr>
<td>tour</td>
<td>Tournament size</td>
<td>2</td>
</tr>
<tr>
<td>pc</td>
<td>Crossover probability</td>
<td>0.8</td>
</tr>
<tr>
<td>pm</td>
<td>Mutation probability</td>
<td>0.2</td>
</tr>
</tbody>
</table>
B. Experiments on Real-World Networks

In order to compare the convergence of CMA-IM with its two variants CGA-IM and MA-IM, we set the parameter maxgen larger than the original setting. However, the parameters in each pair of comparisons remain the same.

Firstly, the comparison between CMA-IM and its variant CGA-IM is made to show the effectiveness of the local search procedure. We test CMA-IM and CGA-IM on two real-world networks, the Dolphin social network and the NetHEPT network. In these experiments, we set maxgen as 100 for the Dolphin network and 1,000 for the NetHEPT network. The results are shown as the green and blue lines in Figs. 5 and 6.

The green and blue lines in Fig. 5 show the 2-hop influence spread obtained by CMA-IM and CGA-IM with generation increasing from 1 to 100 on the Dolphin social network. And the green and blue lines in Fig. 6 show the 2-hop influence spread obtained by CMA-IM and CGA-IM with generation increasing from 1 to 1,000 on the NetHEPT network. For the small Dolphin social network, when the generation is up to 100, the 2-hop influence spread produced by CMA-IM and CGA-IM does not differ a lot from each other. Both of them can reach an optimal solution. CMA-IM with local search can converge within 50 generations while CGA-IM without local search needs more generations. However, for the large NetHEPT network, when the generation is up to 100, the 2-hop influence spread produced by CMA-IM and CGA-IM can evolve to a better solution efficiently within 50 generations. These results demonstrate that local search can speed up the convergence and produce a higher quality solution, especially when the search space is large.

Next, the comparison between CMA-IM and its variant MA-IM is made to illustrate the effectiveness of the network clustering step. We test CMA-IM and MA-IM on two real-world networks, the Dolphin network and the NetHEPT network. The parameter maxgen is set as 100 for the Dolphin network and 1,000 for the NetHEPT network. The results are shown as the green and red lines in Figs. 5 and 6.

The green and red lines in Fig. 5 show the 2-hop influence spread obtained by CMA-IM and MA-IM with generation increasing from 1 to 100 on the Dolphin social network. The green and red lines in Fig. 6 show the 2-hop influence spread obtained by CMA-IM and MA-IM with generation increasing from 1 to 1,000 on the NetHEPT network. In this paper, we apply network clustering step to narrow down the search space of the seed nodes. Table 4 gives the number of candidates resulted after reducing the search space. For the Dolphin social network, we take all the five communities as significant communities. The search space is not reduced obviously, so MA-IM achieves a similar result as that of the CMA-IM. However, CMA-IM can converge within 50 generations while MA-IM needs more than 60 generations. For the NetHEPT network, the influence spread of CMA-IM is much better than that of MA-IM because the search space is reduced nearly 50 times. When the generation increases up to 1,000, MA-IM still cannot achieve the optimal solution. However, CMA-IM only needs less than 50 generations. Therefore, the combination of the network clustering and the memetic algorithm is effective that can improve the performance of our algorithm apparently.

Finally, the comparisons between CMA-IM and other six state-of-the-art algorithms are made. We compare the influence

![FIGURE 5](image1)

**FIGURE 5** Comparisons between CMA-IM and its variants CGA-IM and MA-IM in terms of convergence on the Dolphin social network, respectively.

![FIGURE 6](image2)

**FIGURE 6** Comparisons between CMA-IM and its variants CGA-IM and MA-IM in terms of convergence on the NetHEPT network, respectively.

| TABLE 4 | The information about communities and candidates of the three networks. |
|----------|------------------|------------------|------------------|
| NETWORK  | DOLPHIN | NetGRQC | NetHEPT |
| Nodes   | 62      | 5242   | 15233   |
| Seeds   | 10      | 30     | 30      |
| Communities | 5     | 392    | 1820    |
| Significant communities | 5 | 35    | 50 |
| Candidates | 47 | 227   | 315     |
spread and the running time of each algorithm with the number of seeds increasing on the three real-world networks. In these experiments, the parameter maxgen is set as 50. Figs. 8 to 10 show the influence spread of seven algorithms on the three networks under the IC model whose x-axis represents the seed set size and y-axis represents the influence spread estimated by running Monte-Carlo simulation for 10,000 times. And Fig. 11 shows the running time of seven algorithms whose x-axis represents seven algorithms on the three networks and y-axis represents the running time of each algorithm in log scale. The percentages below about influence spread are computed when the seed set size is 30 (The seed set size is 10 for the case of the Dolphin network).

Fig. 7 is an illustration for the community structure and the ultimate seeds of the Dolphin network. The Dolphin network clusters into five communities, the nodes of the same shape belong to the same community. The color of nodes from dark to light corresponds to the degree of nodes from large to small. The ten bold nodes are the seeds generated by CMA-IM.

![Figure 7](image)

**Fig. 7** An illustration for the community structure and the ultimate seeds of the Dolphin network. The Dolphin network clusters into five communities, the nodes of the same shape belong to the same community. The color of nodes from dark to light corresponds to the degree of nodes from large to small. The ten bold nodes are the seeds generated by CMA-IM.

Fig. 8 shows the influence spread on the Dolphin network. From Fig. 8, we can see that the network is partitioned into five communities by BGLL algorithm and the nodes of the same shape belong to the same community. The color of nodes from dark to light corresponds to the degree of nodes from large to small. The bold nodes are the ultimate seeds generated by CMA-IM. The network is low scale and connected sparsely, and we set the seed set size ranging from 1 to 10 and set the propagation probability $p$ as 0.1.

Fig. 8 shows the influence spread on the Dolphin network. From Fig. 8, we can see that the differences in the influence spread of seven algorithms are not obvious. The result of CELF greedy algorithm is the best and CMA-IM essentially matches CELF. The results of CMA-IM, CMA-HClustering and CMA-SLPA are extremely close. Compared with other heuristics, CMA-IM is 2.3%, 4.6% and 21.4% better than Degree centrality, PageRank and the baseline method Random, respectively.

**Fig. 8** Influence spread of different algorithms on the Dolphin network ($N = 62, M = 159$, and $p = 0.1$).
In Fig. 7, we can see that our generated seeds belong to five communities while the nodes with top 10 degree belong to three larger communities. It shows that selecting many top degree nodes in large communities may not influence more nodes. Clustering network into communities and selecting a number of top degree nodes in each significant community can get efficient candidate nodes. For the running time, Fig. 11 shows that the CELF is quite slow, CMA-IM is one order of magnitude better than CELF. CMA-IM, CMA-HClustering and CMA-SLPA have close running time. The heuristic algorithms outperform CMA-IM in running time while their performances are poor.

In Fig. 9, the influence spread increases more and more gently with selecting more and more top degree nodes. This shows CMA-IM can reduce the overlapping influence spread and find more influential nodes. For the running time, when the network grows larger, the low effectiveness of CELF becomes apparent. It takes hours to find 30 seeds while CMA-IM only needs minutes.

Fig. 10 shows the influence spread on the NetHEPT network. From Fig. 10, we can see that the influence spread produced by CMA-IM almost matches that of CELF with 1.5% lower. The results of the three community-based algorithms are still close to each other. CMA-IM is 12.5%, 13.2% and 173.5% higher than Degree, PageRank and Random, respectively. When looking at the running time, CELF takes two orders of magnitude longer time than CMA-IM. From the comparison between the running time of CMA-IM on NetGRQC and NetHEPT, we can see that the running time on NetHEPT is close to NetGRQC due to the network clustering step which reduces the search space of candidate nodes efficiently.

As summarized from the experimental comparisons, the proposed memetic algorithm plays an important role in speeding up the convergence and finding the promising solutions in a low running time. For the running time, the Random has the best performance. The Degree centrality, PageRank and Random also perform better than CMA-IM. However, the Degree centrality, PageRank and Random cannot provide a seed set with good
quality. Although the CELF method can provide a reliable seed set for influence maximization, it is not scalable for large-scale networks. The proposed CMA-IM algorithm can solve the problem of the influence maximization both effectively and efficiently on social networks with different sizes.

VI. Conclusion

In this paper, an efficient memetic algorithm for information maximization has been proposed. The community property has been incorporated to reduce the search space of seed nodes effectively. Then a problem-specific memetic algorithm has been proposed to optimize the 2-hop influence spread which can estimate the influence spread of a node set effectively. In the memetic algorithm, we design a problem-specific population initialization method and a similarity-based local search procedure, which can accelerate the convergence of the algorithm. The similarity between nodes is taken into consideration to solve the problem of influence overlapping. The experiments on three real-world networks illustrate that the proposed CMA-IM algorithm has a good performance in terms of the effectiveness and efficiency on social networks with different sizes.

Due to the increasing scale of the social networks, there is a need to make influence maximization algorithms more efficient. As a possible future work, we will consider to investigate how to extend CMA-IM to a parallel framework. We will also develop our CMA-IM algorithm to other information cascade models such as the linear threshold model and the weighted cascade model.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (Grant nos. 61273317, 61422209, 61473215), the National Program for Support of Top-notch Young Professionals of China, the Specialized Research Fund for the Doctoral Program of Higher Education (Grant no. 20130203110011) and the Fundamental Research Fund for the Central Universities (Grant no. K501512053).

References

Learning User and Product Distributed Representations Using a Sequence Model for Sentiment Analysis

Tao Chen and Ruifeng Xu  
Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen, CHINA

Yulan He  
School of Engineering and Applied Science, Aston University, Birmingham, UK

Yunqing Xia  
Search Technology Center Asia, Microsoft, Beijing, CHINA

Xuan Wang  
Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen, CHINA

Abstract—In product reviews, it is observed that the distribution of polarity ratings over reviews written by different users or evaluated based on different products are often skewed in the real world. As such, incorporating user and product information would be helpful for the task of sentiment classification of reviews. However, existing approaches ignored the temporal nature of reviews posted by the same user or evaluated on the same product. We argue that the temporal relations of reviews might be potentially useful for learning user and product embedding and thus propose employing a sequence model to embed these temporal relations into user and product representations so as to improve the performance of document-level sentiment analysis. Specifically, we first learn a distributed representation of each review by a one-dimensional convolutional neural network. Then, taking these representations as pre-trained vectors, we use a recurrent neural network with gated recurrent units to learn distributed representations of users and products. Finally, we feed the user, product and review representations into a machine learning classifier for sentiment classification. Our approach has been evaluated on three large-scale review datasets from the IMDB and Yelp. Experimental results show that: (1) sequence modeling for the purposes of distributed user and product representation learning can improve the performance of document-level sentiment classification; (2) the proposed approach achieves state-of-the-art results on these benchmark datasets.

Corresponding Author: Ruifeng Xu  
(Email: xuruifeng@hitsz.edu.cn).

1. Introduction

Sentiment analysis aims to detect opinions (or polarities) expressed regarding a given subject or topic from text [1]. With the rapid growth of social media platforms such as microblogging services, social networking sites and short messaging services, people increasingly share their views and opinions online. As such, sentiment analysis has attracted much attention since opinions or sentiments detected from text are potentially useful for downstream applications including recommender systems [2], social network analysis [3], market forecasting [4] and the prediction of political topics [5].

Traditionally, researchers focused on identifying the polarity of text based on language clues extracted from the textual content of reviews [6], [7], [8], [9]. Many recommendation and review sites offer a wealth of information beyond mere ratings, such as opinion holders (hereafter, users) who expressed their views and target entities (hereafter, products) that received the reviews. It is often observed that a lenient user might give higher rating than a critical user even if they post an (almost) identical review, while popular products are likely to receive more praises than less popular ones. The distributions of polarity ratings over reviews written by
different users or written for different products are often skewed in the real world [10]. Tang et al. reported that sentiment ratings from the same user (or towards the same product) are more consistent than those from different users (or towards different products) [11]. As such, it motivated researchers to exploit user or product information in sentiment analysis.

Some approaches extracted user, product and review features based on the Bag-of-Words assumption, which were then subsequently incorporated into different machine learning classifiers [12], [13]. Others took advantage of topic models in order to capture the interest distribution of users and the content distribution for products [14].

Recently, deep neural networks have been used for distributed representation learning [15], [16] and have shown promising results in sentiment analysis [17], [18], [19]. It is possible to learn the distributed representation of a user or a product, which essentially captures semantic information contained in the reviews posted by the user or via product evaluation. In such a representation, a user or product is represented as a dense and real-valued vector. In previous studies, user, product and review information is incorporated into a purposely-built neural network model in order to learn distributed representations of users and products for the purposes of document-level sentiment classification [11], [20]. However, existing studies have ignored the temporal order of reviews that a user posted or a product received. We argue that the temporal relations of reviews are potentially helpful for learning user and product embeddings. For example, a product that receives positive reviews initially might be more likely to get positive reviews later on. Sequence models, such as recurrent neural network (RNN), are effective in learning temporal information, and have achieved excellent performance on tasks with a focus on temporal sequences [21].

In this paper, therefore, we propose a sequence modeling-based neural network approach to embed temporal relations of reviews into the categories of distributed user and product representations (hereafter, user embeddings and product embeddings for short) learning for the sentiment classification of reviews, in which reviews written by one user or evaluated on one product are considered as a temporal-ordered sequence. Using a document-level composition model, which is a one-dimensional convolutional neural network (1d-CNN), each review is first represented by a review embedding. Then, all the reviews by each user are sorted in a temporal order and every review is labeled by its respective rating score. We call such a...
sequence a user review sequence. This can be easily converted into a review embedding sequence by replacing each review text by its corresponding review embedding. Product review sequences and product review embedding sequences are constructed in a similar way where each sequence contains a set of temporal-ordered reviews for a specific product. We subsequently train a separate RNN with gated recurrent unit (RNN-GRU) [22], which has been shown effective in several sequence-based tasks [23], on the constructed user and product review sequences respectively. Continuous vectors generated in the penultimate layer of the model are regarded as user (or product) embeddings, and they capture important global clues such as user preferences, product qualities and temporal relations of reviews. These embeddings are further integrated with continuous review embeddings in a machine learning classifier for sentiment classification.

We evaluate the effectiveness of our approach empirically using three large-scale review datasets from the IMDB and Yelp. We compare our results with a wide range of baselines including word2vec [15], recursive neural networks [17], paragraph vector [16], user product neural network (UPNN) [11] and a state-of-the-art recommendation algorithm JMARS [14]. Experimental results show that (1) sequence modeling for distributed user and product representation learning can improve the performance of document-level sentiment classification; and (2) the proposed approach achieves state-of-the-art results on these benchmark datasets.

The main contributions of our work are summarized below:

- We formulate user and product representation learning as a sequence modeling problem and successfully employ a RNN-GRU neural network to embed temporal relations of reviews into user and product embeddings.
- We propose a novel approach which combines a neural network model and a traditional machine learning classifier for sentiment classification.
- We conduct extensive experiments on three benchmarking datasets and show that our approach achieves state-of-the-art results for document-level sentiment classification.

The rest of this paper is organized as follows: we discuss related work in Section 2; we then present our approach in Section 3; experimental setup and evaluation results are reported in Section 4; finally, Section 5 concludes the paper and outlines future research directions.

2. Related Works

Sentiment analysis is the field of study that analyzes people’s opinions, sentiments, evaluations, attitudes, and emotions from written languages. These can include reviews, forum discussions, blogs, news, comments or any other types of documents [7]. One of the most prominent tasks in sentiment analysis is sentiment classification.

A large body of work in sentiment classification focused on exploring supervised machine learning approaches using various types of features. Early work [1] trained supervised classifiers including Naïve Bayes, maximum entropy, and support vector machines (SVM) on the Bag-of-Words features for document-level sentiment classification. It was found that although supervised classifiers outperformed human-produced baselines, they did not perform well when compared to the traditional topic-based classification task. Turney [6] proposed to use point-wise mutual information (PMI) to calculate the sentiment orientation (positive or negative) of a word or phrase. The average sentiment orientation scores of all words or phrases in a document determines the overall sentiment orientation of the document. Mullen and Collier [24] used PML features derived from a knowledge base, topic proximity and syntactic relation features to train SVM. Goldberg and Zhu [25] proposed a graph-based semi-supervised learning algorithm to improve sentiment classification performance using unlabeled reviews. Nakagawa et al. [26] presented a dependency tree based method and conditional random fields (CRF) with hidden variables for sentiment classification of Japanese and English subjective sentences. Bickstaffe and Zukerman [27] proposed a hierarchical classifier algorithm that accounted for the inter-class similarity of tagged sentiment-bearing texts for document-level multi-class sentiment classification. Kiritchenko et al. [28] presented a supervised statistical text classification approach leveraging a variety of sentiment features derived from high-coverage tweet-specific sentiment lexicons. These lexicons are automatically generated from tweets with sentiment-word hashtags and from tweets with emoticons. Gangemi et al. [29] implemented a model and a tool to detect opinion holders and topics of opinionated sentences by using a heuristic graph mining approach that relied on a machine reader for the semantic web. Xu et al. [30] explored the use of cross-lingual resources for opinion mining for resource poor languages by using multi-kernel SVM and transfer learning.

Knowledge-based [31] and linguistic pattern based [32] techniques are also popular because of their accessibility and economy. Popular sources of affect words or multiword expressions for sentiment classification include the WordNet-Affect [33], SentiWordNet [34] and SenticNet [35].

2.1 Distributed Representation Learning for Sentiment Classification

Feature engineering approaches are labor intensive and time consuming. The distributed representation proposed by Hinton et al. [36] is a low-dimensional real value vector for text representation. This kind of representation is effective for capturing syntactic and semantic relationships. With the rapid development of deep neural networks and parallel computing, automatically learning representation of knowledge attracts much research interest in recent years.

Models for learning distributed representations of knowledge have been proposed at different granularity levels, including word sense level [37], [38], [39], word level [15], [40], [41], [42], phrase level [22], [43], [44], sentence level [17], [19], [45], discourse level [46] and document level [16].

For sentiment classification, Maas and Ng [47] used a probabilistic model of documents to learn semantically focused word vectors, and evaluated word vectors in two sentiment analysis tasks. Tang et al. [48] encoded sentiment information of text together with context of words in a number of neural networks
with tailored loss functions to automatically derive sentiment embeddings from massive texts. These sentiment embeddings can be used as word features for sentiment analysis without ad hoc feature engineering. Chen et al. [49] combined a WordNet [50] glosses composition model and a context clustering model to learn word sense embeddings which can be used in sentiment analysis tasks.

Socher et al. [17] proposed a recursive neural tensor network (RNTN) for semantic compositionality over a sentiment treebank which pushed the binary classification accuracy on the Stanford sentiment treebank from 80% up to 85.4%. Kalchbrenner et al. [18] proposed a dynamic convolutional neural network (DCNN) to handle input sentences with varying length and induced a feature graph over a sentence that is capable of explicitly capturing short and long-range relations. Kim [19] presented two simple CNN models with little hyperparameter tuning which were trained on pre-trained word vectors for sentence-level classification tasks. Tang et al. [51] introduced a gated recursive neural network to learn continuous document representations for sentiment classification.

2.2 User and Product Modeling for Sentiment Classification
In recent years, there has been growing interests in incorporating the user and product information for sentiment analysis of product reviews. Seroussi et al. [13] presented a nearest-neighbor collaborative approach for training user-specific classifiers whose outputs were subsequently combined with user similarity measurement for sentiment inference from text. Li et al. [12] used the user, product and review features as a three-dimension tensor, and employed tensor factorization techniques to alleviate the data sparsity problem. Gao et al. [52] referred to user- or product-specific sentiment polarity biases as user leniency and product popularity, respectively. They built a model that automatically computed user leniency and product popularity for sentiment classification. Diao et al. [14] proposed a probabilistic model based on collaborative filtering and topic modeling to capture user and product features for sentiment classification. Li et al. [53] incorporated textual topic and user-word factors with supervised topic modeling. Zhang et al. [54] formalized the phrase level sentiment polarity labeling problem in a convex optimization framework, and designed iterative updating algorithms for leveraging review-level sentiment classification techniques to boost the performance of phase-level sentiment polarity labeling. Tang et al. [11], [20] incorporated user, product and review information into a purposely-built neural network model to learn distributed representations of users and products for document-level sentiment classification.

2.3 Sequence Modeling for Sentiment Analysis
Although sentiment classification has been mostly approached as a binary or multi-class classification problem, some studies considered sentiment analysis as modeling of sentiment flow throughout a document, using sequence modeling approaches for sentiment detection. Mao and Lebanon [55] developed a variant of CRF for sentiment flow prediction. Liu and Zhou [56] decomposed a sentence into a series of sub-sequences using a hidden CRF, and determined the sentence-level polarity by classifying within sub-sequences and by fusing the obtained sub-sequence polarities.

Sequence modeling has often been used for fine-grained sentiment analysis. It aims to detect the subjective expressions in a text and to characterize their intensity and sentiment as well as to identify the opinion holder and the target, or topic, of the opinion [57]. Johansson and Moschitti [58] demonstrated relational features derived from dependency-syntactic and semantic role structures are useful for sentiment analysis. Yang and Cardie [59] proposed a semi–CRF-based approach relaxing the Markovian assumption inherent to CRFs and operated at the phrase level rather than the token level, allowing the incorporation of phrase-level features. Isory and Cardie [60] applied deep RNNs to the task of opinion expression extraction formulated as a token-level sequence-labeling task.

In this paper, we proposed to learn user and product embeddings from the temporal ordered reviews written by a user or evaluated on a product respectively using a variant of RNN as a sequence learning model combining user, product and review content information.

3. Methodology
In this section, we present our approach for learning user and product distributed representations using a sequence model for sentiment analysis. An overview of the approach is shown in Fig. 1. We first describe the document composition model which produces the distributed representation of each review document (Section 3.1). Afterwards, we introduce the sequence model for embedding temporal relations of reviews into user and product representations learning (Section 3.2). Finally, we describe the sentiment classification model which encodes user, product and review information (Section 3.3).

3.1 Modeling Reviews with Multi Filter 1d-CNN
We describe our approach for learning review embeddings from review content and its rating with a document composition model (multi filter 1d-CNN).

3.1.1 Training Objective
For reviews with respect to $1−K$ rating scales (sentiment strength scores), the training objective of the document composition model is to minimize the ranking loss below:

$$\sum_{d \in T} \max \{0, 1 - g(d) + g(d')\}$$

where $d$ is a review document in the training set $T$ with a certain rating from 1 to $K$ (positive sample); $d'$ is another review in $T$ with a rating different from $d$ (negative sample); $g(\cdot)$ is a scoring function which represents the whole neural network architecture without the last classification layer; $g(d)$ and $g(d')$ are the score of positive and negative sample, respectively. For each review in the training set, we expect $g(d)$ to be approximating 1, $g(d')$ to be approximating 0,
and $g(d)$ to be larger than $g(d')$ by a margin of 1 after training the neural network.

3.1.2 Neural Network Architecture

The multi-filter 1d-CNN, with its architecture shown in Fig. 2, is used to learn review embeddings. It takes reviews of varying lengths as input and produces fixed-length vectors as output.

Before training, a distributed representation for each word (often referred to as a word embedding [41]) in all reviews is generated. It can be initialized randomly or taken from other pre-trained word embeddings. All the word embeddings are stacked in a matrix $M \in \mathbb{R}^{d \times |V|}$, where $d$ is the dimension of word embedding and $|V|$ is the size of word vocabulary. In the input layer, embeddings of words in the current training review are taken from $M$. In order to handle input reviews of varying lengths, the maximum length of reviews that the network handles is set to $m$. Shorter reviews are padded with zero vectors. Then, dropout regularization, introduced by Hinton et al. [61], is used to control over-fitting.

In the convolution layer, a filter moves on the word embeddings to perform one-dimensional convolution. The idea behind the one-dimensional convolution is to take the dot product of the filter vector $w \in \mathbb{R}^d$ with each $n$-gram in the review $r$ to obtain another sequence $c$. As the filter moves on, many sequences are generated. We call them feature sequences. The $i$-th feature sequence $c_i \in \mathbb{R}$ is generated as follows:

$$c_i = f(w \cdot t_i + b)$$  \hspace{1cm} (2)

where $n$ is the size of filter; $b \in \mathbb{R}$ is a bias term and $f(\cdot)$ is a point-wise non-linear activation function, such as the hyperbolic tangent (tanh), sigmoid or rectified linear units (ReLU); $t_i$ refers to word embeddings of $r$ in current filter window. Many features are combined into a feature map, $F_j \in \mathbb{R}^{n \times m-1}$, as defined below:

$$F_j = [c_1, c_2, \ldots, c_{n-1}],$$  \hspace{1cm} (3)

In the pooling layer, a max-overtime pooling operation [62], which forces the network to capture the most useful local features produced by the convolutional layers, is applied over $F_j$. We further add activation functions to incorporate element-wise nonlinearity. The outputs of multiple filters are concatenated in the merge layer. After another dropout process, a fully connected softmax layer output the probability distribution over labels from multiple classes.
3.1.3 Training

The softmax function is an activation function often implemented at the final layer of a network used for classification [63]. It predicts the probability distribution over classes given the input. In this $K$-classes task, given an input review document $d$ and its representation vector $v_d$, a conditional tag probability $p(y = i | v_d)$ by applying a softmax operation is defined as follows:

$$p(y = i | v_d) = \frac{e^{w_i v_d + b_i}}{\sum_{k=1}^{K} e^{w_k v_d + b_k}}$$  \hspace{1cm} (4)

where $i \in 1, ..., K$, $w$ and $b$ are the parameters of the merge layer.

The network is trained to minimize the mean of the negative log-likelihood of the prediction of this model under a given target distribution, which is computed as follows:

$$\text{mean} \{ \forall i \in 1, ..., K \mid - \log p(y = i \mid v_d) \}.$$  \hspace{1cm} (5)

The training error propagates back to fine-tune the parameters of the networks and the input word embeddings in each mini-batch, and Adagrad [64] is performed as the gradient descent optimization method. The vectors generated in the merge layer are regarded as review embeddings which capture the semantic features of the input reviews, to some degree.

3.2 Learning User and Product Embeddings from Temporal-ordered Reviews with Sequence Modeling

In this section, we explain how the temporal information about a user or a product, i.e., temporal order of the reviews written by one user or evaluated on one product, is captured in our approach. As shown in Fig. 1, the obtained review embeddings are grouped by the identical user or the identical product, respectively. In each group, review embeddings with corresponding ratings (labels) are treated as a temporal-ordered sequence ordered by their posted time to create a user review embedding sequence and a product review embedding sequence, respectively. Then, these two sequences are fed to a sequence model to learn user embeddings and product embeddings, respectively. We present here a RNN-GRU neural network for sequence modeling.

3.2.1 Recurrent Neural Network with Gated Recurrent Unit

As illustrated in Fig. 3, a RNN with GRU for learning user or product embeddings is a neural network that takes a user or product review embedding sequence $(x_1, ..., x_T)$ as input, consisting of a hidden GRU $h_t$ and an optional output $y$. $T$ is the last time step. Time refers to the idea that a sequence has a notion of order, e.g., a review sequence ordered by posted time.

At each time step $t$, the hidden state $h_t$ of the GRU is computed based on the previous hidden state $h_{t-1}$ and the input at the current step $x_t$:  

$$h_t = f(U x_t + W h_{t-1})$$  \hspace{1cm} (6)

where $U$ and $W$ are parameter matrices of the network, and $f(\cdot)$ is a non-linear activation function. It may be as simple as an element-wise logistic sigmoid function and as complex as a long short-term memory (LSTM) unit [65] or GRU [22].

Both LSTM and GRU are RNN architectures explicitly designed to deal with vanishing gradients problem [66] and efficiently learn long-range dependencies through a gating mechanism. A GRU has two gates, a reset gate $r$ and an update gate $z$. They are computed as follows:

$$I = \delta(U x_t + W h_{t-1})$$  \hspace{1cm} (7)

$$z = \delta(U^z x_t + W^z h_{t-1})$$  \hspace{1cm} (8)
3.3 Sentiment Classification

Up to now, we have embedded variable-length reviews, user review sequences and product review sequences in a fixed-dimensional space which can capture semantic relation in review content, enabling more efficient similarity comparison. A major advantage of embeddings in a fixed-dimensional space is that a wide variety of machine learning algorithms are then applicable.

We concatenate the three embeddings into a single embedding for each review in the training set. Reviews written by the same user or evaluated on the same product use the same user embedding and product embedding, respectively. Then, the concatenated embeddings are used to train a SVM [67] for sentiment classification.

We have also tried to concatenate the three embeddings and use a neural network with a softmax function at the final layer for classification. Experiment results show that training SVMs with the concatenated embeddings achieves better performance than using a neural framework model (detailed in section 4.3.3).

4. Experiment

We conduct experiments to evaluate the performance of the proposed approach for document-level sentiment classification on three datasets. In this section, we describe the experimental setup and baseline methods followed by the discussion of results.

4.1 Experimental Setup

The three large-scale datasets ¹ include one movie review dataset from IMDB developed by Diao et al. [14], and two restaurant review datasets from the Yelp Dataset Challenge² in 2013 and 2014 developed by Tang et al. [11]. The statistics of the three datasets are summarized in Table 1. The rating scale used in the IMDB dataset is 1-10, whereas the rating scale used in Yelp 2013 and 2014 datasets is 1-5. We train the rating predictor on the training set, tune parameters on the development set and evaluate on the test set.

We conduct sentiment classification on these three datasets. Following Tang et al.’s work [11], we use accuracy as the evaluation metric to measure the overall sentiment classification performance, and use mean absolute error (MAE) and root mean squared error (RMSE) as the evaluation metrics to measure the divergences between predicted and ground truth sentiment ratings. MAE and RMSE are computed as follows:

\[
\text{MAE} = \frac{1}{N} \sum_{i=1}^{N} | \text{gold}_i - \text{predicted}_i | \quad (12)
\]

\[
\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (\text{gold}_i - \text{predicted}_i)^2} \quad (13)
\]

¹Available at: http://ir.hit.edu.cn/~dytang/paper/acl2015/dataset.7z

²http://www.yelp.com/dataset_challenge
where \( \text{gold} \) is the true rating for the \( i \)-th review, \( \text{predicted} \) is the predicted rating and \( N \) is the total number of reviews in our test set. Smaller values indicate more accurate prediction, and hence a better model.

For training CNN, we use: ReLU as activation function, filter windows of 3, 4, 5 with 100 feature maps each, Adadelta decay parameter of 0.95, dropout rate of 0.5, the size of initial word vectors of 300. For training RNN-GRU, we use: ReLU as activation function, dropout rate of 0.25, time distributed fully connected layer with softmax, Adam as stochastic optimization method, categorical cross-entropy as loss function, 300 hidden units.

### 4.2 Baseline Methods

We benchmark the following baseline methods as having been previously used in [11] for document-level sentiment classification: Trigram, TextFeature, Trigram+UPF, TextFeature+UPF, JMARS, AvgWordvec+SVM, SSWE+SVM, Paragraph Vector, RNTN+Recurrent, UPNN (no UP), and UPNN (full).

Trigram, TextFeature, Trigram+UPF, and TextFeature+UPF are hand-crafted text features based methods. In Trigram, unigrams, bigrams and trigrams are used as features to train a SVM classifier. In TextFeature, word/character \( n \)-grams, sentiment lexicon features, and negation features are used [28]. In Trigram+UPF and TextFeature+UPF, user-leniency features [52] and corresponding product features from training data are concatenated with the features in Trigram and TextFeature, respectively.

JMARS is a state-of-the-art recommendation algorithm proposed by Diao et al. [14], in which user and aspects features of a review are used with collaborative filtering and topic modeling.

AvgWordvec+SVM, and SSWE+SVM are two distributed word representation based methods. In AvgWordvec+SVM, word embeddings are learned by word2vec [15], and the mean vector of each word in a document is used as document representation. In SSWE+SVM, sentiment-specific word embeddings (SSWE) [68] are learned, and document representation are generated by max/min/average pooling. Both of them train SVM classifiers for document sentiment classification.

Paragraph Vector, RNTN+Recurrent, UPNN (no UP), and UPNN (full) are distributed document representation based methods. Paragraph Vector is an unsupervised framework proposed by Le and Mikolov [16]. In RNTN+Recurrent, sentence representations are learned by recursive neural tensor network (RNTN) [17], and document representations are composed with RNN. UPNN uses a CNN to compose reviews written by the same user or written on the same product for sentiment classification of documents [11]. In this method, the temporal order of reviews is ignored. UPNN (no UP) uses review content only without considering user and product information and UPNN (full) uses review content, user and product information. Tang et al. report that UPNN (full) achieves state-of-the-art performances on IMDB, Yelp 2013 and 2014 datasets in their work [11].

### 4.3 Results

#### 4.3.1 Overall Comparison

Table 2 shows the results achieved on the IMDB, Yelp 2013 and Yelp 2014 datasets. The best results are highlighted in bold face. The methods marked with a star use both user and product information in addition to review content, while others only use review texts. The results of the top 11 approaches have been previously reported in [11]. The bottom two approaches are ours. We present the results obtained using our approach without user and product information, denoted as our approach (no UP), or with review content, user and product information taken into account, denoted as our approach (full).

It is observed that for the four text feature based methods, Trigram gives comparable performance to TextFeature which relies on hand-crafted features. Incorporating user and product features improves the classification performance of Trigram and TextFeature on all the three metrics and across all the three datasets. Topic model based method, JMARS, performs similarly compared to text feature based methods on the IMDB dataset, but gives worse results on the other two datasets.

Distributed word representation based methods and distributed document representation based methods can automatically generate features for classifier training. However, two word embedding based methods and Paragraph Vector perform worse than hand-crafted features based methods. RNTN+Recurrent and UPNN (no UP) give mixed results compared to hand-crafted features based methods. UPNN (full) outperforms all the other baselines by using CNN composition and taking into account of all three types of information, including review content, users and products.

Our sequence modeling based approach outperforms all the baselines by a large margin. Without using user and product information, our approach (no UP) gives relative improvements

### TABLE 1 Statistics of IMDB, Yelp 2014 and Yelp 2013 datasets. \#review, \#user and \#prod denote the number of reviews, users and products, respectively.

<table>
<thead>
<tr>
<th>DATASET</th>
<th>TRAINING SET</th>
<th>DEVELOPMENT SET</th>
<th>TEST SET</th>
<th>CLASSES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>#REVIEW</td>
<td>#USER</td>
<td>#PROD</td>
<td></td>
</tr>
<tr>
<td></td>
<td>#REVIEW</td>
<td>#USER</td>
<td>#PROD</td>
<td></td>
</tr>
<tr>
<td></td>
<td>#REVIEW</td>
<td>#USER</td>
<td>#PROD</td>
<td></td>
</tr>
<tr>
<td>IMDB</td>
<td>67,426</td>
<td>1,310</td>
<td>1,635</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8,381</td>
<td>1,310</td>
<td>1,635</td>
<td></td>
</tr>
<tr>
<td></td>
<td>9,112</td>
<td>1,310</td>
<td>1,635</td>
<td>10</td>
</tr>
<tr>
<td>YELP 2013</td>
<td>62,522</td>
<td>1,631</td>
<td>1,633</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7,773</td>
<td>1,631</td>
<td>1,633</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8,671</td>
<td>1,631</td>
<td>1,633</td>
<td>5</td>
</tr>
<tr>
<td>YELP 2014</td>
<td>183,019</td>
<td>4,818</td>
<td>4,194</td>
<td></td>
</tr>
<tr>
<td></td>
<td>22,745</td>
<td>4,818</td>
<td>4,194</td>
<td></td>
</tr>
<tr>
<td></td>
<td>25,399</td>
<td>4,818</td>
<td>4,194</td>
<td>5</td>
</tr>
</tbody>
</table>
of 15.6% in accuracy, 12.3% in MAE, and 8.7% in RMSE compared to UPNN (full) on the IMDB dataset. On Yelp 2013 and 2014 datasets, the ranges of relative improvements are 6.7–8.1% in accuracy, 13.0–14.8% in MAE, and 10.8–12.2% in RMSE, respectively.

With the user and product information, our approach (full) improves upon UPNN (full) by 12.2% in accuracy, 13.1% in MAE, and 9.4% in RMSE on the IMDB dataset. On Yelp 2013 and 2014 datasets, the ranges of relative improvements are 5.1–7.2% in accuracy, 11.9–14.9% in MAE, and 9.9–11.5% in RMSE, respectively. It shows that the use of sequence modeling for distributed user and product representation learning is effective in improving the performance of document-level sentiment analysis.

We also observe that with user and product information, our approach (full) gives superior performance compared to our approach (no UP) on all the three datasets, with the relative improvements ranging between 2.4–4.3% in accuracy, 3.9–5.8% in MAE and 2.3–2.7% in RMSE, respectively. We apply two-sample $t$-test on the experimental results of these two approaches, the result indicates statistical significance on all the three datasets ($p < 0.05$). All these results show that incorporating user and product embeddings can boost the performance compared to the models without using them.

### 4.3.2 Review Sequence Approach (RNN-GRU) vs. Unordered Set Approach (CNN)

We design a CNN version of our approach which ignores the temporal order of the reviews when learning user or product embeddings. For user embedding learning, we aggregate all the reviews written by the same user and concatenate them into a long review with temporal order ignored before feeding it to the CNN. We do it similarly for product embedding learning by aggregating all the reviews about the same product without considering their temporal relation. Other components of our proposed approach, such as learning review embeddings and concatenating review content, user and product embeddings for SVM training, remain the same. We call this variant unordered set approach since all the reviews of the same user or the same product are considered as an unordered set. Our original approach using RNN-GRU for user and product embedding learning is termed as review sequence approach.

![Fig 4](image_url) Experimental results of review sequence approach (RNN-GRU) and unordered set approach (CNN) on the Yelp 2013 dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>ACCURACY</th>
<th>MAE</th>
<th>RMSE</th>
<th>ACCURACY</th>
<th>MAE</th>
<th>RMSE</th>
<th>ACCURACY</th>
<th>MAE</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRIGRAM</td>
<td>0.399</td>
<td>1.147</td>
<td>1.783</td>
<td>0.569</td>
<td>0.513</td>
<td>0.814</td>
<td>0.577</td>
<td>0.487</td>
<td>0.804</td>
</tr>
<tr>
<td>TEXTFEATURE</td>
<td>0.402</td>
<td>1.134</td>
<td>1.793</td>
<td>0.556</td>
<td>0.520</td>
<td>0.845</td>
<td>0.572</td>
<td>0.490</td>
<td>0.800</td>
</tr>
<tr>
<td>TRIGRAM+UPF*</td>
<td>0.404</td>
<td>1.132</td>
<td>1.764</td>
<td>0.570</td>
<td>0.491</td>
<td>0.803</td>
<td>0.576</td>
<td>0.471</td>
<td>0.789</td>
</tr>
<tr>
<td>TEXTFEATURE+UPF*</td>
<td>0.402</td>
<td>1.129</td>
<td>1.774</td>
<td>0.561</td>
<td>0.509</td>
<td>0.822</td>
<td>0.579</td>
<td>0.476</td>
<td>0.791</td>
</tr>
<tr>
<td>JMARS*</td>
<td>N/A</td>
<td>1.285</td>
<td>1.773</td>
<td>N/A</td>
<td>0.699</td>
<td>0.985</td>
<td>N/A</td>
<td>0.710</td>
<td>0.999</td>
</tr>
<tr>
<td>AVGWORDVEC+svm</td>
<td>0.304</td>
<td>1.361</td>
<td>1.985</td>
<td>0.526</td>
<td>0.568</td>
<td>0.898</td>
<td>0.530</td>
<td>0.562</td>
<td>0.893</td>
</tr>
<tr>
<td>SSWE+svm</td>
<td>0.312</td>
<td>1.347</td>
<td>1.973</td>
<td>0.549</td>
<td>0.529</td>
<td>0.849</td>
<td>0.557</td>
<td>0.523</td>
<td>0.851</td>
</tr>
<tr>
<td>PARAGRAPH VECTOR</td>
<td>0.341</td>
<td>1.211</td>
<td>1.814</td>
<td>0.554</td>
<td>0.515</td>
<td>0.832</td>
<td>0.564</td>
<td>0.496</td>
<td>0.802</td>
</tr>
<tr>
<td>RNTN+RECURRENT</td>
<td>0.400</td>
<td>1.133</td>
<td>1.764</td>
<td>0.574</td>
<td>0.489</td>
<td>0.804</td>
<td>0.582</td>
<td>0.478</td>
<td>0.821</td>
</tr>
<tr>
<td>UPNN (NO UP)</td>
<td>0.405</td>
<td>1.030</td>
<td>1.629</td>
<td>0.577</td>
<td>0.485</td>
<td>0.812</td>
<td>0.585</td>
<td>0.483</td>
<td>0.808</td>
</tr>
<tr>
<td>UPNN (FULL)*</td>
<td>0.435</td>
<td>0.979</td>
<td>1.602</td>
<td>0.596</td>
<td>0.464</td>
<td>0.784</td>
<td>0.608</td>
<td>0.447</td>
<td>0.764</td>
</tr>
<tr>
<td>OUR APPROACH (NO UP)</td>
<td>0.468</td>
<td>0.903</td>
<td>1.487</td>
<td>0.624</td>
<td>0.413</td>
<td>0.713</td>
<td>0.624</td>
<td>0.410</td>
<td>0.704</td>
</tr>
<tr>
<td>OUR APPROACH (FULL)*</td>
<td>0.488</td>
<td>0.851</td>
<td>1.451</td>
<td>0.639</td>
<td>0.395</td>
<td>0.694</td>
<td>0.639</td>
<td>0.394</td>
<td>0.688</td>
</tr>
</tbody>
</table>
shows that the temporal relations indeed characterize users and products better.

4.3.3 Comparison of Traditional Classifiers with a Pure Neural Framework

There are two main steps in our approach: firstly, learning user, product and review embeddings; secondly, feeding these embeddings into SVM training. We also design a pure neural version of our approach, in which SVM is replaced by a fully connected layer with the softmax function and hence sentiment classification can be performed in the process of representation learning.

We present the experimental results of this pure neural framework and our approach using four different classifiers in Fig. 5, where Neural indicates this pure neural framework. LR, NB and Linear indicate our approach with SVM replaced by logistic regression [69], Naive Bayes, and LIBLINEAR [70], respectively.

It is observed that neural framework outperforms logistic regression and Naive Bayes, but gives worse results compared to LIBLINEAR and SVM on all the three metrics. The best performance is obtained using SVM, which outperforms the neural framework by a large margin. This shows that for the datasets experimented here, it is better to separate representation learning from sentiment classifier training.

4.3.4 Comparison with Different Sequence Models

We have also experimented with different sequence models, including RNN and LSTM, for learning user and product representations. Fig. 6 shows the results of using RNN, LSTM and GRU as a sequence model on the Yelp 2013 dataset. It can be observed that GRU outperforms LSTM, which in turn gives better performance compared to RNN on all the three metrics. It is because both GRU and LSTM, having more complicated hidden units, offer better composition capability than RNN. GRU has fewer parameters to train compared to LSTM and thus generalizes better than LSTM.

5. Conclusion

This paper has presented a sequence modeling based neural network approach for document-level sentiment analysis. The approach employs RNN-GRU to learn user and product embeddings from the temporal ordered review documents. These embeddings, together with review embeddings learned by a CNN, are used to train SVMs for sentiment classification. We have conducted extensive experiments on three review datasets using three evaluation metrics. Empirical results show that our approach achieves the state-of-the-art performance on all these datasets. We have found that (1) modeling reviews as sequences rather than unordered sets boosts the performance of user and product representation composition; (2) concatenating review, user and product embeddings for training SVMs for sentiment classification gives superior results compared to a pure neural framework and beats the best results reported so far. Evaluations on three large-scale datasets show that the proposed method performs better than several strong baseline methods which regard reviews as unordered set.

In future work, we plan to explore other sequence learning model, such as bidirectional RNN, bidirectional LSTM and gated feedback RNN for sentiment analysis. We will also explore other methods in learning user and product embeddings and investigate the feasibility of using these embeddings for a wide range of tasks such as product recommendation and product sales prediction.

Acknowledgments

This work is supported by the National Natural Science Foundation of China 61370165, National 863 Program of China 2015AA015405, Shenzhen Development and Reform Commission Grant No. [2014]1507, Shenzhen Peacock Plan Research Grant KQCX20140521144507925 and Shenzhen Foundational Research Funding JCYJ2015062514254370.

References

I. Introduction

The information age has brought along an explosion of Big Data [1], from multiple sources in every aspect of our lives: human activity signals from wearable sensors and personal devices, experiments in particle discovery research and stock market data systems are few examples. Big social data analysis [2] is the area of research focusing on collecting, examining and processing large multi-modal and multi-source datasets in order to discover patterns/correlations and extract information from the Social Web. This is usually accomplished through the use of computationally expensive supervised and unsupervised machine learning algorithms that learn from the available data (e.g., Support Vector Machines-SVMs [3], Artificial Neural Networks-ANNs, [4], k-Nearest Neighbors-kNN [5], and Random Forests-RF [6]) that are not able to...
handle current data volumes [7]. Parallel approaches have been proposed in order to boost processing speeds [8] but this clearly requires technologies that support distributed computations.

Extreme learning machine (ELM) [9] is an emerging learning paradigm, presenting an efficient unified solution to generalized feed-forward neural networks. Unlike ANNs, however, ELM cannot be easily parallelized, due to the presence of a pseudo-inverse calculation [10]. Therefore, this paper aims to find a reliable method to realize a parallel implementation of ELM that can be applied to large datasets typical of Big Data problems. An example of parallel ELM implementation for regression based on the MapReduce framework can be found in [11], while [12] provides a parallel ensemble method for an Online Sequential ELM variant.

Several technologies that exploit multiple levels of parallelism (e.g., multi-core, many-core, GPU, cluster, etc.) are currently available [13]–[16]. Spark [17] in combination with cloud computing [18], [19] is a state-of-the-art framework for high performance parallel computing designed to efficiently deal with iterative computational procedures that recursively perform operations over the same data, such as supervised machine learning algorithms.

Apart from building supervised learning models efficiently and with scalable algorithms, another important issue in Big Data is how to effectively and efficiently assess the performance of a predictive model. Data-driven models exploit non-parametric inference, where it is expected that an effective model would stem directly from the data, without any assumption on the model family nor any other information that is external to the dataset itself [20]. With the advent of the Big Data era, this approach has increasingly gained popularity, with the belief that effective predictive models, with the desired accuracy, can be generated by simply collecting larger volumes of data (see, as an example, [21] for some insights on this provocative and inexact but, unfortunately, widespread belief).

Statistical Learning Theory (SLT) addresses the problem of assessing the performance of a predictive model, by trying to find necessary and sufficient conditions for non-parametric inference to build predictive models from data [22]–[26] or, in the language of SLT, learn an optimal model from data. For a long time, SLT was considered only a theoretical, albeit very sound and deep, statistical framework, without any real applicability to practical problems [27]. With important advances in this field over the last decade [28], it has been shown that SLT can provide practical answers, at least when targeting the inference of data-driven models for classification purposes [29], [30].

II. Related Work

In recent years, opinions and sentiments of the masses have increasingly been publicly conveyed through social networks, web communities, blogs, wikis, and other online collaborative media. This has deeply changed the way people share knowledge and communicate experiences. As a result, the distillation of useful information from the massive amount of opinions is a key tool for marketers trying to create a product, brand, or organization image or identity in the minds of their customers.

This has led to an in-depth development of the field of sentiment analysis, which deals with information retrieval and knowledge discovery from text using data mining and natural language processing (NLP) techniques [31].

Main approaches to big social data analysis can be broadly grouped into two categories: knowledge-based techniques [32] and statistical methods [33]. While the former mainly leverage on ontologies [34], lexicons [35], semantic networks [36], or patterns [37], the latter are gradually shifting to the adoption of ELM, deep learning and convolutional neural network (CNN).

In particular, Tang et al. [38] developed a CNN-based approach to obtain word embeddings for words popularly used in tweets, then fed them into the network for sentiment analysis. A deep CNN for sentiment detection in short text was also proposed by Santos et al. [39]. The approach based on Sentiment Specific Word Embeddings [40] considers word embeddings based on a sentiment corpora: this means including more affective clues than regular word vectors, thus producing a better result. The importance of studying the granularity of emotions in social networks is underlined by the new project ‘Reactions’, developed by Facebook: instead of only the Like button, a more complete choice of emotions is proposed (e.g., ‘love’, ‘fun’, ‘anger’, ‘surprise’, ‘sadness’). A similar approach was adopted by Poria et al. [41], which extracted features from short texts, based on the activation values of an inner layer of a deep CNN, and used these for multimodal sentiment analysis.

III. Preliminary Definitions

Let us first focus on the binary classification problem [4], [22], [42]. Let \( \mathcal{X} \subseteq \mathbb{R}^d \) and \( \mathcal{Y} \in \{ \pm 1 \} \) be, respectively, the input and the output spaces. We consider a set of labeled independent and identically distributed (i.i.d.) data \( \mathcal{D} = \{ (x, y) \} \) of size \( n \), where \( z \in \{1, \ldots, n\} \) and \( (x, y) \in \mathcal{X} \times \mathcal{Y} \), sampled from an unknown distribution \( \mu \). As we are targeting at Big Data problems [7], [43], [44], we will focus on the case where \( n \) is very large. For later use we also define two modified training sets: \( \mathcal{D}^+ = \{ (1, \ldots, n, z_1, \ldots, z_d) \} \) and \( \mathcal{D}^- = \{ (1, \ldots, n, z_1', \ldots, z_d') \} \), where, respectively, the \( i \)-th element is removed or replaced by another sample [25].

A learning algorithm \( \mathcal{A}_H \), characterized by a set of hyperparameters \( H \) that must be tuned, maps \( \mathcal{D} \) into a function \( f : \mathcal{A}_{D, H} \) from \( \mathcal{X} \) to \( \mathcal{Y} \). In particular, \( \mathcal{A}_H \) allows designing \( f \in \mathcal{F}_H \) and the class of functions \( \mathcal{F}_H \), which is generally unknown (and depends on \( H \)) [26], [28], [30]. The accuracy of a function \( f : \mathcal{A}_{D, H} \) in representing the hidden relationship \( \mu \) is measured with reference to a loss function \( \ell(x, f, z) \in [0, 1] \) [25]. In particular, since we are dealing with binary classification problems, the loss function (called the hard loss) simply counts the number of missclassified examples [22], [45]: \( \ell_h(x, f, z) = 1[y(x) \neq 0] \in \{ 0, 1 \} \).

The quantity which we are interested in is the generalization error [22], [30], namely the error that a model will perform on new data generated by \( \mu \) and previously unseen \( L(f) = E_x \ell_h(x, f, z) \). Unfortunately, since \( \mu \) is unknown, \( L(f) \) cannot be computed and, consequently, must be estimated.
Two common empirical estimators are the empirical $L_{emp}(f) = 1/n \sum_{i=1}^{n} \ell(f, z)$ [22] and leave-one-out $L_{loo}(f) = 1/n \sum_{i=1}^{n} \ell(f(\mathcal{A}(i)); x_i, z_i)$ [46] errors.

IV. Extreme Learning Machines

The ELM approach [9], [47], [48] was introduced to overcome problems posed by the back-propagation training algorithm [49]; specifically, potentially slow convergence rates, the critical tuning of optimization parameters, and the presence of local minima that call for multi-start and re-training strategies. In this section, we recall conventional ELM and then adapt it to the Big Data framework. ELM was originally developed for the single hidden layer feedforward neural networks [50], [51] and then generalized in order to cope with cases where ELM is not neuron alike:

$$f(x) = \sum_{j=1}^{k} w_j g_j(x),$$

(1)

where $g_j: \mathbb{R}^d \rightarrow \mathbb{R}, j \in \{1, \ldots, h\}$ is the hidden layer output corresponding to the input sample $x$ and $w$ is the output weight vector between the hidden layer and the output layer.

In our case, the input layer has $d$ neurons and connects to the hidden layer (having $h$ neurons) through a set of weights $w_j \in \mathbb{R}^d, j \in \{1, \ldots, h\},$ (2)

the $j$-th hidden neuron embeds a bias term, $v_j, j \in \{1, \ldots, h\},$ (3)

and a nonlinear activation function, $\phi: \mathbb{R} \rightarrow \mathbb{R}$. Thus the neuron’s response to an input stimulus, $x$, is:

$$\phi(v_j \cdot x + v_j), j \in \{1, \ldots, h\}. \quad (4)$$

Note that Eq. (4) can be further generalized to include a wider class of functions [50], [51], [52]; therefore, the response of a neuron to an input stimulus $x$ can be generically represented by any nonlinear piecewise continuous function characterized by a set of parameters. In ELM, these parameters $(v_j$ and $v_j)$ are set randomly. A vector of weighted links, $w \in \mathbb{R}^h$, connects the hidden neurons to the output neuron without any bias. The overall output function, $f(x)$, of the network is:

$$f(x) = \sum_{j=1}^{h} w_j \phi(v_j \cdot x + v_j). \quad (5)$$

It is convenient to define an activation matrix, $V \in \mathbb{R}^{h \times h}$, such that the entry $V_{ij}$ is the activation value of the $j$-th hidden neuron for the $i$-th input pattern. The $V$ matrix is:

$$V = \begin{bmatrix} \phi(v_1 \cdot x_1 + v_1) & \cdots & \phi(v_h \cdot x_1 + v_h) \\ \vdots & \ddots & \vdots \\ \phi(v_1 \cdot x_k + v_1) & \cdots & \phi(v_h \cdot x_k + v_h) \end{bmatrix} = \phi^T(x). \quad (6)$$

In the ELM model, the quantities $\{v_j, v_j\}$ in Eq. (4) are set randomly and are not subject to any adjustment, and the quantity $w$ in Eq. (5) is the only degree of freedom. Hence, the training problem reduces to minimization of the convex cost:

$$w^* = \arg \min_w \|Vw - y\|,$$

(7)

A matrix pseudo-inverse yields the unique $L_2$ solution, as proven in [51], [53]:

$$w^* = V^+y. \quad (8)$$

The simple, efficient procedure to train the ELM therefore involves the following steps:

1) Randomly generate hidden node parameters (in or case $v_j$ and bias $v_j$) for each hidden neuron;
2) Compute the activation matrix $V$, of Eq. (6);
3) Compute the output weights by solving the pseudo-inverse problem of Eq. (8).

Despite the apparent simplicity of the ELM approach, the crucial result is that even random weights in the hidden layer endow a network with notable representation ability. Moreover, the theory derived in [53] proves that regularization strategies can further improve the approach’s generalization performance. As a result, the cost function of Eq. (7) is augmented by a regularization factor as follows:

$$w^* = \arg \min_w \|Vw - y\| + \lambda \|w\|,$$

(9)

where $\|w\|$ can be any suitable norm of the output weights [53]. A common approach is then to use the $L_2$ regularizer:

$$w^* = \arg \min_w \|Vw - y\| + \lambda \|w\|,$$

(10)

and consequently the vector of weights $w^*$ is then obtained as follows:

$$w^* = \left((V^T V + \lambda I)^{-1} V^T y, \quad (11)$$

where $I \in \mathbb{R}^{h \times h}$ is an identity matrix.

V. ELM for Big Data on Spark

Spark [17] is a state-of-the-art framework for high performance memory parallel computing designed to efficiently deal with iterative computational procedures that recursively perform operations over the same data [14], [17], [54]. One recent solution for Big Data analytics is the use of cloud computing [19], [55], [56], which makes available hundreds or thousands of machines to provide services such as computing and storage.

Various cluster management options are available for running Spark [57]. In this work, we chose to deploy Spark in a Hadoop cluster. The selected Hadoop architecture was composed of $N_M$ slave machines and two additional machines that run as masters: one for controlling HDFS and the other for resource management.
The software packages installed on each machine were Hadoop 2.7.1 and Spark 1.5.1. In order to exploit this architecture, we had to modify the ELM formulation to cope with the main computational issues of ELM:

(I) computing the matrix V of Eq. (6);

(II) finding the solution of $w^*$ of Eq. (11).

The main idea behind the Spark technology is that we have to reduce access to the disk as much as possible and make as much computation as possible in memory. Moreover, since Spark is designed to efficiently deal with iterative computation-intensive problems that recursively perform operations over the same data, it may not be efficient to compute the solution in the form of Eq. (11).

Hence, let us start from issue (II). Instead of solving the problem of Eq. (9) with the approach of Eq. (11), let us adopt a Stochastic Gradient Descent (SGD) algorithm. The SGD algorithm is a very general optimization algorithm, which is able to solve a problem efficiently in the following form:

$$\min_{w \in \mathbb{R}^d} \sum_{i=1}^m C(w, \phi(x_i), y_i) + \lambda R(w),$$

where $R(w)$ is a regularizer [58]–[61] and $C(w, \phi(x), y)$ is a convex relaxation of the Hard Loss Function [45]. $\lambda$ balances the tradeoff between the over- and underfitting tendency of the algorithm. Based on the choice of $R(w)$ and $C(w, \phi(x), y)$, we can retrieve different algorithms. If $R(w) = 0$ and $C(w, \phi(x), y) = (w^T \phi(x) - y)^2$, we get the ELM formulation of Eq. (9). Other possible choices are: SVM [62], Regularized Least Squares [63], Least Squares SVM [64], Logistic Regression [65], Lasso [66], Elastic Net [61], etc. In Table 1 we report a series of possible choices of $R(w)$ and $C(w, \phi(x), y)$.

The SGD algorithm for optimizing Problem (12) is reported in Algorithm 1 [44]. In Algorithm 1, $\tau$ and $n_{iter}$ are parameters related with the speed of the optimization algorithms. Therefore, usually $\tau$ and $n_{iter}$ are set based on the experience of the user. In any case $\tau$ and $n_{iter}$ can be seen as other regularization terms as $\lambda$ since they are connected with the early stopping regularization technique [66], [67].

Algorithm 1 is well-suited for implementation in Spark and many of these tools are already available in MLlib [68]. Basically, the implementation of Algorithm 1 reported in Algorithm 2 is an application of three functions: a filter (for the gradients that require an IF condition in Table 1), a map for the computation of the gradient and a reduction function for the sum of each single gradient.

The main problem of Algorithm 2 is the computation and storage of $V$. If $k \ll d$ it means that $V \in \mathbb{R}^{k \times n}$ will be much smaller than the dataset which belongs to $\mathbb{R}^{d \times n}$. In this case, it is more appropriate to compute it before the SGD algorithms

### Table 1: Possible Regularizers and Convex Approximations of the Hard Loss Function.

<table>
<thead>
<tr>
<th>(A) REGULARIZERS</th>
<th>$R(w)$</th>
<th>$\frac{\partial}{\partial w} R(w)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>$</td>
<td>w</td>
</tr>
<tr>
<td>R2</td>
<td>$</td>
<td>w</td>
</tr>
<tr>
<td>R3</td>
<td>$</td>
<td>w</td>
</tr>
<tr>
<td>R4</td>
<td>$\eta</td>
<td>w</td>
</tr>
<tr>
<td>R5</td>
<td>$</td>
<td>w</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(B) CONVEX APPROXIMATIONS OF THE HARD LOSS FUNCTION</th>
<th>$C(w, \phi(x), y)$</th>
<th>$\frac{\partial}{\partial \phi} C(w, \phi(x), y)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>$w^T \phi(x) - y$</td>
<td>$sign(w^T \phi(x) - y) x_u$</td>
</tr>
<tr>
<td>L2</td>
<td>$(w^T \phi(x) - y)^2$</td>
<td>$2 (w^T \phi(x) - y) x_u$</td>
</tr>
<tr>
<td>L3</td>
<td>$\max(0,1 - y, w^T \phi(x))$</td>
<td>$-y x_u$ if $y w^T \phi(x) \leq 1$</td>
</tr>
<tr>
<td>L4</td>
<td>$\max(0,1 - y, w^T \phi(x))^2$</td>
<td>$-2 y (1 - y w^T \phi(x)) x_u$ if $y w^T \phi(x) \leq 1$</td>
</tr>
<tr>
<td>L5</td>
<td>$\ln(\frac{1}{1 + \exp(-y w^T \phi(x))})$</td>
<td>$-y \frac{x_u}{1 + \exp(-y w^T \phi(x))}$</td>
</tr>
</tbody>
</table>

### Algorithm 1: SGD for Eq. (12).

**Input**: $D_m$, $\lambda$, $\tau$, $n_{iter}$

**Output**: $w$

1. Read $D_m$.
2. Compute $V$ (Eq. (6));
3. $w = 0$;
4. for $t \leftarrow 1$ to $n_{iter}$ do
5. $w = w - \frac{\tau}{\sqrt{t}} \sum_{i=1}^n C(w, \phi(x), y) + \lambda R(w)$;
6. return $(w, b)$;

### Algorithm 2: SGD for Eq. (12) on Spark.

**Input**: $D_m$, $\lambda$, $\tau$, $n_{iter}$

**Output**: $w$

1. Read $D_m$;
2. Compute $V$ (Eq. (6));
3. $w = 0$;
4. for $t \leftarrow 1$ to $n_{iter}$ do
5. $w = w - \frac{\tau}{\sqrt{t}} \sum_{i=1}^n C(w, \phi(x), y) + \lambda R(w)$;
6. return $(w, b)$;
VI. Performance Assessment and Uncertainty Quantification

The selection of the optimal hyperparameters of a predictive model is the fundamental problem of STL, which is still the target of current research [28], [30], [69]–[72]. The approaches can be divided in two large families: Resampling methods; like Hold Out, Cross Validation, and the Non-parametric Bootstrap [30], [71], [73], [74], and more recent In-Sample methods; like the class of function-based methods [30] (based on the VC-Dimension [62], Rademacher Complexity (RC) [23], [24], [75], [76], PAC-Bayes Theory [77], [78]), and algorithm-based methods [26] (based on Compression Bounds [79], and Algorithmic Stability (AS) Theory [25], [80]).

Resampling methods [30], [81] are favored by practitioners because they work well in many situations and allow the application of simple statistical techniques for estimating the quantities of interest. Some examples of resampling methods are the well-known k-Fold Cross Validation (KCV), the Leave-One-Out (LOO), and the Non-parametric Bootstrap (BTS) [71], [74], [82].

In-Sample methods [30], [81], instead, allow the exploitation of a whole set of available data for both training the model and estimating its generalization error, thanks to the application of rigorous statistical procedures [25], [28], [78].

For more details about the advantages and disadvantages of the different methods one can refer to [26], [29], [30].

A. Resampling Methods

These techniques rely on a similar idea: the original dataset \( \mathcal{D}_n \) is resampled once or many \( (n_r) \) times, with or without replacement, to build three independent datasets called training, validation, and test sets respectively, \( \mathcal{L}_i', \mathcal{V}_i, \) and \( \mathcal{T}_i' \), with \( r \in \{1, \ldots, n_r\} \). Note that \( \mathcal{L}_i' \cap \mathcal{V}_i = \emptyset, \mathcal{L}_i' \cap \mathcal{T}_i' = \emptyset, \) and \( \mathcal{V}_i \cap \mathcal{T}_i' = \emptyset \). Then, in order to select the best set of hyperparameters \( \mathcal{H} \) in a set of possible ones \( \mathcal{R} = \{ \mathcal{H}_1, \mathcal{H}_2, \ldots \} \) for the algorithm \( \mathcal{H}_n \) or, in other words, to perform the Model Selection (MS), we have to apply the following procedure:

\[
\mathcal{H}^*: \min_{\mathcal{H} \in \mathcal{R}} \frac{1}{n_r} \sum_{r=1}^{n_r} \hat{L}_{\text{MSE}}(\mathcal{A}_{\mathcal{D}_i,n}, \mathcal{V}_i). \tag{13}
\]

Since the data in \( \mathcal{L}_i' \) are i.i.d. from the one in \( \mathcal{V}_i \), the idea is that \( \mathcal{H}^* \) should be the set of hyperparameters which allows one to achieve a small error on a dataset that is independent from the training set.

The uncertainty quantification, instead, is performed as follows:

\[
L(\mathcal{A}_{\mathcal{D}_i,n}, \mathcal{H}_r) \leq \frac{1}{n} \sum_{i=1}^{n} \hat{L}_{\text{MSE}}(\mathcal{A}_{\mathcal{D}_i,n}, \mathcal{V}_i, \mathcal{T}_i) + \sqrt{\frac{\pi}{2n}}, \tag{14}
\]

where the bound holds with probability \((1 - e^{-c})\). Note that after the best set of hyperparameters is found, one can select the best model by training the algorithm with the whole dataset \( \mathcal{A}_{\mathcal{D}_i,n}, \mathcal{H}_r \), [69], [83]. Moreover, since the data in \( \mathcal{L}_i' \cup \mathcal{V}_i \) are i.i.d. with respect to \( \mathcal{T}_i \), we have that \( \hat{L}_{\text{MSE}}(\mathcal{A}_{\mathcal{D}_i,n}, \mathcal{V}_i, \mathcal{T}_i) \) is an unbiased estimator of \( L(\mathcal{A}_{\mathcal{D}_i,n}, \mathcal{H}_r) \). Then, we can use any concentration result, like the Hoeffding inequality [84], for bounding the bias between the expected value and its empirical estimator.

Note, also, that we get the hold-out method [30] if \( n_r = 1 \), if \( l, v \), and \( t \) are set a priori such that \( n = l + v + t \) and if the resample procedure is performed without replacement. For implementing the complete KCV, instead, we have to set \( n_r \leq \binom{n}{l} \left( \frac{n-l}{a} \right) = (k-2)(n/k), v = n/k, \) and \( t = n/k \) and the resampling must be done without replacement [30], [69], [71]. Finally, for implementing the Non-parametric Bootstrap, \( l = n \) and \( \mathcal{L}_i' \) must be sampled with replacement from \( \mathcal{D}_n \), while \( \mathcal{V}_i \) and \( \mathcal{T}_i' \) are sampled without replacement from the
sample of $\mathcal{D}_n$ that has not been sampled in $\mathcal{L}$ [30], [74]. Note that for the Non-parametric Bootstrap procedure $n_s \leq \left( \frac{2n}{n} - 1 \right)$.

It is worthwhile noting that the only hypothesis needed in order to rigorously apply the resampling technique is the i.i.d. hypothesis on the data in $\mathcal{D}$, and that all these techniques work for any deterministic algorithm.

B. In-Sample Methods

For the In-Sample methods, two subfamilies of techniques are identified: the class of function-based ones and algorithm-based ones [26]. The difference between the two classes is that the function-based techniques require the knowledge of $\mathcal{F}_H$ and thus, cannot be applied to some algorithms (e.g., the kNN algorithm), while the algorithm-based techniques can be applied to any deterministic algorithm without any additional knowledge. Both subfamilies, like the resampling methods, require the i.i.d. hypothesis.

One of the most powerful techniques in the class of function-based techniques is based on the Rademacher Complexity [23], [30]. In particular, for any bounded loss function $\ell_i(f, z) \in [0,1]$ it is possible to prove that the following bound holds with probability $(1 - 2e^{-\gamma})$ [85]:

$$L(f) \leq \hat{L}_{\text{emp}}(f; \mathcal{D}_n) + \hat{R}_s(\mathcal{F}_H) + 3\sqrt{\frac{X}{2n}}, \forall f \in \mathcal{F}_H$$

(15)

where

$$\hat{R}_s(\mathcal{F}_H) = \mathbb{E}_s \sup_{f \in \mathcal{F}_H} \frac{2}{n} \sum_{i=1}^{n} \sigma_i \ell_i(f, z), \quad \sigma_{i \in \{1,\ldots,n\}} \in \{-1,1\}, \quad \mathbb{P}\{\sigma_i = 1\} = \mathbb{P}\{\sigma_i = -1\} = \frac{1}{2}. \tag{16}$$

Therefore, based on the Structural Risk Minimization principle [22], one can design a series of function classes of increasing size, $\mathcal{F}_H = \{\mathcal{F}_H, \mathcal{F}_{H1}, \ldots\}$ with $\mathcal{F}_H \subseteq \mathcal{F}_{H1} \subseteq \cdots$, so to compute at the same time both the MS and the uncertainty quantification:

$$f^*, \mathcal{F}_H: L(f^*) \leq \min_{f \in \mathcal{F}_H} \left[ \hat{L}_{\text{emp}}(f; \mathcal{D}_n) + \hat{R}_s(\mathcal{F}_H) + \sqrt{\frac{9X}{2n}} \right]. \tag{17}$$

Unfortunately, the quantity of Eq. (16) cannot be computed if we do not know $\mathcal{F}_H$. Moreover, for many algorithms it is impossible to define $\mathcal{F}_H$ [26]. Algorithm-based techniques circumvent this problem through the concept of Algorithmic Stability [25], [26], [86]. In particular, for any bounded loss function $\ell_i$ it is possible to prove that the following bounds hold with probability $(1 - \delta)$ [25]:

$$L(f) \leq \hat{L}_{\text{emp}}(f; \mathcal{D}_n) + \sqrt{\frac{1}{2n\delta} + \frac{3\beta_{\text{emp}}}{\delta}}, \quad \text{(18)}$$

$$L(f) \leq \hat{L}_{\text{uw}}(f; \mathcal{D}_n) + \sqrt{\frac{1}{2n\delta} + \frac{3\beta_{\text{uw}}}{\delta}}, \quad \text{(19)}$$

where $f : \mathcal{A}_{\mathcal{D}, \mathcal{H}}$ and

$\beta_{\text{emp}}(\mathcal{A}_{\mathcal{H}, n}) = \mathbb{E}_{x, z} \left[ \ell_i(A_{\mathcal{D}, \mathcal{H}}, z) - \ell_i(A_{\mathcal{D}, \mathcal{H}}, z) \right], \quad \beta_{\text{uw}}(\mathcal{A}_{\mathcal{H}, n}) = \mathbb{E}_{x, z} \left[ \ell_i(A_{\mathcal{D}, \mathcal{H}}, z) - \ell_i(A_{\mathcal{D}, \mathcal{H}}, z) \right].$

The bounds of Eqs. (18) and (19) are polynomial bounds in $n$ (not very tight indeed when $n$ is small) while $\beta_{\text{emp}}$ and $\beta_{\text{uw}}$ are two versions of Hypothesis Stability which are able to take into account both the properties of the algorithm and the property of the distribution that has generated the data $\mathcal{D}$, [25], [26]. It is possible to improve the bounds of Eqs. (18) and (19) by exploiting a stronger notion of algorithmic stability, known as the Uniform Stability. In particular, the following bounds hold with probability $(1 - \delta)$:

$$L(f) \leq \hat{L}_{\text{emp}}(f; \mathcal{D}_n) + 2\beta' + (4n\beta' + 1) \sqrt{\frac{\ln \left( \frac{1}{\delta} \right)}{2n}}, \quad \text{(20)}$$

$$L(f) \leq \hat{L}_{\text{uw}}(f; \mathcal{D}_n) + \beta'' + (4n\beta'' + 1) \sqrt{\frac{\ln \left( \frac{1}{\delta} \right)}{2n}}, \quad \text{(21)}$$

where $f : \mathcal{A}_{\mathcal{D}, \mathcal{H}}$ and

$$\beta'(A_{\mathcal{H}, n}) = \left[ \ell(A_{\mathcal{D}, \mathcal{H}}, \cdot) - \ell(A_{\mathcal{D}, \mathcal{H}}, \cdot) \right]_v, \quad \text{(22)}$$

$$\beta''(A_{\mathcal{H}, n}) = \left[ \ell(A_{\mathcal{D}, \mathcal{H}}, \cdot) - \ell(A_{\mathcal{D}, \mathcal{H}}, \cdot) \right]_v. \quad \text{(23)}$$

Unfortunately, the Uniform Stability ($\beta'$ or $\beta''$) is not able to take into account the properties of the distribution that has generated the data $\mathcal{D}$, and is sometimes unable to capture the properties of the algorithm because it deals with a worst-case learning scenario [26]. Nevertheless, all the four stability-based bounds of Eqs. (18), (19), (20), and (21) can be used to select the best set of hyperparameters $\mathcal{H}$ in a set of possible $\hat{\mathcal{S}} = \{\mathcal{H}_1, \mathcal{H}_2, \ldots\}$ for the algorithm $A_{\mathcal{H}}$. In particular, all the bounds are expressed in the form: $L(A_{\mathcal{D}, \mathcal{H}}) \leq \varepsilon(\mathcal{A}_{\mathcal{H}, \mathcal{D}_n, n}, \delta \mathcal{H})$. Thus, in order to perform the MS procedure and uncertainty quantification, we have to aprioriistically assign to each set of hyperparameters a probability $p_{\mathcal{H}}$ (where $\sum_{\mathcal{H}} p_{\mathcal{H}} = 1$) of being chosen during the MS procedure. The algorithmic stability-based MS and uncertainty quantification procedure can then be summarized as follows:

$$A_{\mathcal{D}, \mathcal{H}_n}, \mathcal{H}' : L(A_{\mathcal{D}, \mathcal{H}'}) \leq \min_{\mathcal{H}} \varepsilon(\mathcal{A}_{\mathcal{H}, \mathcal{D}_n, n}, \delta \mathcal{H}). \quad \text{(24)}$$

The procedure of Eq. (24) can be exploited with any algorithm for which it is possible to compute one of the notions of stability.

VII. Computational Issues for Big Data Analytics

Both naive resampling and In-Sample methods are computationally expensive when the number of samples is large [30], [72]. For this reason, we will focus on adapting these techniques to the Big Data context.

A. Bag of Little Bootstraps

The standard Non-parametric Bootstrap procedure requires, $\forall \mathcal{H} \in \{\mathcal{H}_1, \mathcal{H}_2, \ldots\}$, to train many ($n$) models, and is computationally very expensive if $n$ is large. For this reason the Bag of
Little Bootstraps approach [87]–[90] represents an alternative to standard Non-parametric Bootstrap; it considers only $b = n^q$ data, with $q \in [1/2, 1]$, in place of the whole dataset during the creation of the train, validation and test sets. Note that $q \in [1/2, 1]$ is necessary to maintain the statistical property of the procedure. In particular, the Bag of Little Bootstraps [87] consists in sampling $n_{\text{rep}}$ times from $D_n$ without replacement, several datasets $B_i$ with $i \in \{1, \ldots, n_{\text{rep}}\}$, consisting of $b \in \sqrt{n \cdot n}$ samples. Then, each $B_i$ is sampled with replacement $n_{\text{rep}}$ times, in order to derive $L^q_n$ datasets with $f \in \{1, \ldots, n_{\text{rep}}\}$, each consisting of $n$ samples. All the samples of $D_n$, or parts of them, that have not been sampled in $L^q_n$ are used as validation set and test set $V^q_i \subseteq D_n \setminus L^q_n$, $T^q_i \subseteq D_n \setminus L^q_n$ and $V^q_i \cap T^q_i = \emptyset$. Finally, the models are trained on the sets $L^q_n$ and tested on the corresponding $V^q_i$, thus we define the following MS procedure:

$$
\mathcal{H}^* = \min_{\mathcal{H}^*} \frac{1}{n_{\text{rep}}} \sum_{i=1}^{n_{\text{rep}}} \mathbb{E}[\ell_{\text{emp}}(\mathcal{A}_{L^q_n, \mathcal{H}^*}, V^q_i)].
$$

(25)

Note that in order to find $\mathcal{H}^*$ with the procedure of Eq. (25), we have to train a series of models over sets composed by a maximum of $n^q$ distinct samples. This means that the MS strategy, if $n$ is large with respect to $n_{\text{rep}}$ and $n_{\text{rep}}$, scales with $n^q$. Therefore, the procedure scales sub-linearly with respect to $n$, and in the best case scenario, scales with $O(\sqrt{n})$. Analogous to the usual Non-parametric Bootstrap procedure, the uncertainty quantification is performed as follows:

$$
L(\mathcal{A}_{D_n, \mathcal{H}^*}) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}[\ell_{\text{emp}}(\mathcal{A}_{L^q_n, \mathcal{H}^*}, T^q_i)] + \sqrt{\frac{\gamma}{2n}},
$$

(26)

where the best model is obtained by training the algorithm with the whole dataset [83]: $f^* = \mathcal{A}_{D_n, \mathcal{H}^*}$.

Finally, we would like to underline that $\gamma$ balances the tradeoff between accuracy and computational requirements of the statistical procedure [88], [90]. The more $\gamma \rightarrow 1$, the better the MS strategy will perform. Since we deal with Big Data in this paper, we set $\gamma = 1/2$. The application of this approach to ELM is straightforward by noting that the hyperparameters of ELM are $\lambda \in [0, \infty]$ and $h \in \{1, 2, \ldots\}$.

B. Simplified Rademacher Complexity

Now, we show that the Rademacher Complexity of ELM (which employs the general regularization schema of Eq. (12)) can be easily upper bounded. In particular, let us truncate the loss functions such that $\ell_{\text{emp}}(f, z) = \min[1, C(w, \phi(x), y)]$. It is easy to see that $\mathbb{E}[\ell_{\text{emp}}(f, z)] \leq \mathbb{E}[\ell_{\text{emp}}(f, z)]$. Consequently, the generalization error computed with $\mathbb{E}[\ell_{\text{emp}}(f, z)]$ is equal to or less than the one computed with $\ell_{\text{emp}}(f, z)$. By exploiting the bound of Eq. (15) for $\ell_{\text{emp}}(f, z)$ the computation of the empirical error is straightforward and it is possible to prove that the Rademacher Complexity can be upper bounded as follows [85]:

$$
R_n(F_H) = \mathbb{E}[\sup_{f \in F} \frac{2}{n} \sum_{i=1}^{\tau_n} \sigma_i \ell_{\text{emp}}(f, z)] \\
\leq \frac{L}{n} \sqrt{\sum_{i=1}^{\tau_n} \|w^* \phi(x)\|^2},
$$

(27)

where $L$ is the Lipschitz constant characterizing $\ell_{\text{emp}}(f, z)$ and $(w^*, \phi)$ is the solution to the ELM problem of Eq. (10) (or more generally Eq. (12)) for a given $h$ and $\lambda$. $C(w, \phi(x), y)$ and $R(w)$ can be any of the ones reported in Table 1. Note that $h$ and $\lambda$ define the size of the class of functions in ELM [30] and thus, we can plug this result in the procedure of Eq. (17) and obtain a computationally efficient way of assessing the performance and quantifying the uncertainty of ELM. In fact, in order to exploit the procedure of Eq. (17), it is only necessary to train, for each values of $h$ and $\lambda$, the ELM model and to compute the quantity of Eq. (27) which is computationally inexpensive once the ELM has been trained.

C. Simplified Uniform Stability

In this section, we show how to apply the bound based on the Uniform Stability in the Big Data scenario. The bound of Eq. (21), which takes into account the leave-one-out error, is too computationally expensive to compute. Instead, we employ that of Eq. (20). As in Section (VII-B), we use the truncated loss function since for the hard loss function we have trivially that $\beta(\mathcal{A}_H, n) = 1$. Consequently, once the ELM has been trained we can compute the empirical error with the truncated loss. Computing $\beta(\mathcal{A}_H, n)$ is not easy but, thanks to the result of [25], it is possible to upper bound it in the case of ELM as follows:

$$
\beta(\mathcal{A}_H, n) = \mathbb{E}[[\ell_{\text{emp}}(\mathcal{A}_H, \mathcal{H}^*) - \ell_{\text{emp}}(\mathcal{A}_H, \mathcal{H}^*)]_+ I_{n \lambda}] \\
\leq \frac{L}{n} \max \left\{ \|\phi(x)\|, \ldots, \|\phi(x)\| \right\}.
$$

(28)

where $L$ is the Lipschitz constant characterizing $\ell_{\text{emp}}(f, z)$. In this case, $R(w)$ must be $\|w\|_2$, since the bound does not hold for all the $R(w)$ reported in Table 1. Then the application of the procedure of Eq. (24) to ELM becomes straightforward and computationally inexpensive. From Eq. (28) it seems that the Uniform Stability takes into account only the property of ELM through $\lambda$ and $h$ through $\phi$, and not the distribution of the data. In other words, the Uniform Stability upper bound of Eq. (28) is not able to capture the effect of changing the loss.

D. Bag of Little Hypothesis Stabilities

In order to overcome the issues of the Uniform Stability, we exploit the proposal of [26] to estimate the Hypothesis Stability instead. As we will see, this proposal is also well suited for Big Data applications. As for the Uniform Stability, we do not consider the bound of Eq. (19) since it is too computationally expensive. Consequently, we take into account the bound of Eq. (18). In this case, we do not need to exploit the truncated loss function, but can use the hard loss function directly once the ELM model has been trained with a fixed value of $\lambda$. In order to compute the bound of Eq. (18) and perform the procedure of Eq. (28), we need to compute $\beta_{\text{emp}}(\mathcal{A}_H, n)$. We start by making an assumption on the learning algorithm $\mathcal{A}_H$. In particular, we suppose that the Hypothesis Stability does not increase with the cardinality of the training set:

$$
\beta_{\text{emp}}(\mathcal{A}_H, n) \leq \beta_{\text{emp}}(\mathcal{A}_H, \sqrt{n} - 1).
$$

(29)
Affective analogical reasoning can be defined as the intrinsically human capacity to interpret the cognitive and affective information associated with natural language.

We point out that this property is a desirable requirement for any learning algorithm, because in order to be able to prove the learnability in the stability framework, we need that:

$$\lim_{n \to \infty} \beta_{\text{emp}}(\mathcal{A}_H, n) = 0,$$  

(30)

or, in other words, that the impact on the learning procedure of removing or replacing one sample from \(\mathcal{D}_n\) should decrease, on average, as \(n\) grows. Numerous researchers have already studied this property in the past. In particular, it is related to the consistency concept [46]. However, connections can also be identified with the trend of the learning curves of an algorithm [91]. Moreover, such quantity is also strictly linked to the concept of Smart Rule [46]. It is worth underlining that, in many of the above-referenced works, the property of Eq. (29) is proved to be satisfied by many well known algorithms (SVM, Regularized Least Squares and consequently ELM, k-Local Rule with \(k > 1\), etc.).

Let us define \(\hat{\beta}_{\text{emp}}(\mathcal{A}_H, n, \mathcal{D}_n)\) which is an unbiased estimator of \(\beta_{\text{emp}}(\mathcal{A}_H, n)\):

$$\hat{\beta}_{\text{emp}}(\mathcal{A}_H, \sqrt{n} - 1, \mathcal{D}_n) = \frac{1}{\sqrt{n}} \sum_{i=1}^{\sqrt{n}} \left[ \ell(\mathcal{A}_H(z_i, z^*_i), \hat{z}_i) - \ell(\mathcal{A}_H(z_i, z^*_i), \hat{z}^*_i) \right],$$  

(31)

where \(i \in \{1, \ldots, \sqrt{n} - 1\}\). Moreover:

$$\mathcal{D}_i \triangleq \{ z_{(k-1)/\sqrt{n}+1}, \ldots, z_{(k-1)/\sqrt{n} + \sqrt{n}-1} \},$$  

(32)

$$\hat{z}_{k} = z_{(k-1)/\sqrt{n}}.$$  

(33)

By construction we have that \(\hat{\beta}_{\text{emp}}(\mathcal{A}_H, \sqrt{n} - 1, \mathcal{D}_n)\) is an unbiased estimator of \(\beta_{\text{emp}}(\mathcal{A}_H, \sqrt{n} - 1)\):

$$\mathbb{E}_{\mathcal{D}_n} \hat{\beta}_{\text{emp}}(\mathcal{A}_H, \sqrt{n} - 1, \mathcal{D}_n) = \beta_{\text{emp}}(\mathcal{A}_H, \sqrt{n} - 1).$$  

(34)

Since all the quantities in the summations of Eq. (31) are \(\{\pm 1\}\) valued i.i.d. random variables (since they are computed over different sets of data) extracted from a Bernoulli distribution of mean \(\beta_{\text{emp}}(\mathcal{A}_H, \sqrt{n} - 1)\), we have that the following bound holds [84] with probability \((1 - e^{-c})\):

$$\beta_{\text{emp}}(\mathcal{A}_H, \sqrt{n} - 1) \leq \hat{\beta}_{\text{emp}}(\mathcal{A}_H, \sqrt{n} - 1, \mathcal{D}_n) + \frac{x}{\sqrt{2/\sqrt{n}}}.$$  

(35)

Note that plugging Eq. (35) into the bound of Eq. (18) gives a fully empirical bound where all the quantities can be computed from the data [26]. In particular, once the ELM has been trained for a given \(h\) and \(\lambda\), the empirical error, computed with the hard loss function, is trivially computable, while \(\beta_{\text{emp}}(\mathcal{A}_H, \sqrt{n} - 1, \mathcal{D}_n)\) requires the training of many ELMs on a small subset of the data \((\sqrt{n})\), which is computationally inexpensive. Moreover, all these ELMs can be trained in parallel (see Eq. (31)). The application of the procedure of Eq. (24) to ELM then becomes straightforward. Note that, from Eq. (31), the hypothesis stability is able to capture both the property of the algorithm and the property of the distribution that has generated the data [26].

VIII. Affective Analogical Reasoning Dataset

The proposed approach has been tested on two affective analogical reasoning datasets. Affective analogical reasoning can be defined as the intrinsically human capacity to interpret the cognitive and affective information associated with natural language [92]. In particular, we employed two benchmarks, each one composed by 21743 common-sense concepts; each concept is represented according to the AffectiveSpace model [93] and the AffectiveSpace 2 model [94]. Both models are obtained as a vector space representation of the AffectNet network, a semantic network in which common-sense concepts (e.g., ‘read book’, ‘payment’, ‘play music’) are linked to a hierarchy of affective domain labels (e.g., ‘joy’, ‘amazement’, ‘fear’, ‘admiration’). In this way, concepts conveying similar semantic and affective information, e.g., ‘enjoy conversation’ and ‘chat with friends’, tend to fall near each other in the multi-dimensional space. Both AffectNet and AffectiveSpace are publicly available at http://sentic.net. The difference between the two models is the following:

- AffectiveSpace is obtained applying principal component analysis (PCA) on the matrix representation of AffectNet [93].
- AffectiveSpace 2 is obtained applying a refined projection on the matrix representation of AffectNet [94].

In both cases, common-sense concepts are eventually represented by vectors of \(M\) coordinates. This number indicates the dimensionality of the AffectiveSpace and represents the trade-off between efficiency and precision: the bigger is \(M\), the more precisely AffectiveSpace represents AffectNet’s knowledge, but generating the vector space is slower, while the smaller is \(M\), the more efficiently AffectiveSpace can be obtained. As already mentioned, concepts with the same affective orientation are likely to have similar features; i.e., 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 [95]. The Hourglass of Emotions [95] is employed to reason on the disposition of concepts in AffectiveSpace. In the model, affective states are represented by four concomitant but independent dimensions (Pleasantness, Attention, Sensitivity and Aptitude), which determine the intensity of the expressed/perceived emotion. Therefore, a four-dimensional vector can potentially...
synthesize the level of activation of each affective dimension of a concept. Beyond emotion detection, the Hourglass model is also used for polarity detection tasks. Polarity is defined in terms of the four affective dimensions, according to the formula:

$$ p = \frac{1}{\sum_{i=1}^{N} P(c_i), N} [At(c_i) - |S(c_i)| + Ap(c_i)] $$

(36)

where \( P \) is the pleasantness, \( At \) the attention, \( S \) the sensitivity, \( Ap \) the aptitude, \( c \) an input concept, \( N \) the total number of concepts, and 3 the normalization factor (as the Hourglass dimensions are defined as floats \( [-1, 1] \)). In the equation, Attention is taken as absolute value since both its positive and negative intensity values correspond to positive polarity values (e.g., ‘surprise’ is negative in the sense of lack of Attention, but positive from a polarity point of view). Similarly, Sensitivity is taken as negative absolute value since both its positive and negative intensity values correspond to negative polarity values (e.g., ‘anger’ is positive in the sense of level of activation of Sensitivity, but negative in terms of polarity). The publicly available Sentic API (on http://sentic.net/api) was used to obtain for each concept the level of each affective dimension.

According to the Hourglass model, the Sentic API expresses the levels as numbers \( [-1, 1] \), which are eventually mapped into the associated polarity according to Eq. (36). In order to perform a binary classification task for each affective dimension and polarity, the values are then discretized: +1 for positive values and −1 for negative ones.

The experiments eventually involve two tasks:

- Classification of each affective dimension level and polarity detection for concepts expressed according to AffectiveSpace 1 [93];
- Classification of each affective dimension level and polarity detection for concepts expressed according to AffectiveSpace 2 [94].

In both cases, the dimension of the space \( M \) has been set equal to 100.

**IX. Experimental Results**

In this section 1, we show the results of applying the ELMs models described in Section V to the Affective Analogical Reasoning datasets described in Section VIII, where the performance of the models has been assessed by using the MS strategies described in Section VII.

In Tables 2 and 3 we have reported, respectively for AffectiveSpace 1 and AffectiveSpace 2 and for the Pleasantness, the error on the reference set of the ELMs model selected by exploiting regularizer \( \| w \| \), different losses (L1, ..., L5 in Table 1), and different MS strategies (Bag of Little Bootstraps-BLB, Simplified Rademacher Complexity-SRC, Simplified Uniform Stability-SUS, and Bag of Little Hypothesis Stabilities-BLHS). In Table 4, for AffectiveSpace 1, we have reported the time required to build the ELMs model selected by exploiting different losses and different MS strategies. In particular, we reported only the time required for the Pleasantness task.

From Tables 2, 3, and 4 we can state that:

- AffectiveSpace 2 is able to better predict the affective dimensions and polarity with respect to AffectiveSpace 1.
- BLHS is the best method to perform MS since it is the one that more often selects the most accurate model according to the formula.
consistent generalization bounds that can be exploited to these problems, which allows for the obtaining of rigorous and
cal inference frameworks. SLT implements a worst-case approach to the use of the most recent results from SLT. Unlike other statisti-
cal inference frameworks, in the context of big social data analysis. In particular, ELMs on Spark, in order to exploit the benefits of the Spark
architecture, more and more data are becoming available but just a small amount is supervised [96].

Figures 1(a), 1(b) and 1(c) report on the same information for different values of the hyperparameters $h$, $n$, $d$, and $\lambda$. From Figures 1(a), 1(b) and 1(c) it is possible to state that:

- As expected, when $h$ is smaller or comparable to $d$, we have that Algorithm 2 is the one with the best performance.
- When $h$ becomes larger than $d$, the data set to fit into memory; this increases the number of accesses to the disk for Algorithm 2 and consequently, the time needed to execute each iteration. Subsequently, Algorithm 3 becomes more efficient.

X. Conclusion
In this paper, we proposed an efficient implementation of the ELMs on Spark, in order to exploit the benefits of the Spark framework, in the context of big social data analysis. In particular, an approach to support emotion recognition and polarity detection in natural language text has been proposed and evaluated.

We also showed how to carefully assess the performance with the use of the most recent results from SLT. Unlike other statistical inference frameworks, SLT implements a worst-case approach to these problems, which allows for the obtaining of rigorous and consistent generalization bounds that can be exploited for assessing the performance of the ELMs. Thanks to recent advances, as presented in this paper, the computational requirements of these methods have been improved to allow for the scaling to large datasets, which are typical of Big Data applications.

Additional work in this direction is needed. In particular, other big data architectures are available with higher efficiency but lower fault tolerance (e.g., the one based on MPI and OpenMP [18]). It will also be interesting to extend these approaches to a semi-supervised setting since in Big Social Data Analysis more and more data are becoming available but just a small amount is supervised [96].

References
Collaborative Brain-Computer Interface for People with Motor Disabilities

Abstract
This research investigates the effects of collaboration mode, luminance contrast and motor disability on task performance, brain activity and satisfaction of users with motor disabilities who performed a robot control task using a collaborative brain-computer interface (C-BCI) based on steady-state visually evoked potentials (SSVEPs). Users can perform the task by himself/herself (individual mode), working together (simultaneous mode), or taking turns (sequential mode). Fourteen amyotrophic laterals sclerosis (ALS) participants and fourteen able-bodied participants of similar age were recruited from local ALS association and local communities. Results showed that participants in both groups had significantly better task performance and stronger brain activity in collaborative modes than individual mode. High luminance contrast produced significantly better task performance and stronger brain activity than low luminance contrast. There was no significant effect of motor disability between ALS participants and able-bodied participants. The two groups showed very similar task performance and brain activity. These results could provide precious empirical data and invaluable insights to the real-world applicability of the SSVEP-based BCI applications for people with motor disabilities.

I. INTRODUCTION
Brain–computer interface (BCI) is a new interaction medium that allows users to communicate with the external world or control the external equipment without the use of normal pathways of peripheral nerves and muscles [1]. BCIs have shown to have the capability of providing great benefits to people with motor disabilities – increasing their quality of life both physically and spiritually [1]–[4]. BCIs can also reduce the burden oftentimes placed upon the caregiver [5]. Research has shown that both able-bodied users and people with motor disabilities can use BCI systems with acceptable accuracy levels for both communication and device control [4], [6].

Although BCI research has great promise, there are many challenges to overcome [7]–[9]. First, most BCI research focus on speed and accuracy. However, usability is more important for users; especially those with motor disabilities. As BCI is becoming an important tool in neurorehabilitation [10], [11], further research is required to develop more user-friendly BCI-based applications that cater for people with motor disabilities. Secondly, most BCI studies are single-user based and have not explored the integration of BCI into every day normal life, especially to support interactive work such as collaborative work [1], [12]. There has been a general lack of understanding regarding how BCIs should support collaborative work between users with motor disabilities and between users with and without motor disabilities under various task conditions. However, collaborative BCIs (C-BCIs) were shown to bring greater benefits to those with motor disabilities [13]. More studies regarding...
C-BCIs will help to determine how the applications could be extended to real-life situations.

The objective of this research was to investigate the user performance, brain activity, and BCI design preference of people with motor disabilities (e.g., amyotrophic lateral sclerosis, stroke, spinal cord injury, etc.) using an SSVEP-based C-BCI, Brainbot [14], as a test bed.

II. LITERATURE REVIEW

A. Collaborative BCIs

While BCIs hold the promise to restore the communication and control ability of users with motor disability, BCI research has not fully addressed, yet, the social burdens of their disabilities (i.e., interaction with other people). People with motor disability have had little opportunity to work jointly with other people. Meanwhile, very few studies have explored the integration of BCI in normal life [1], [12], especially to support interactive work such as information exchange and collaboration with other people. New BCI applications have recently been validated, such as control of a robotic arm [15], neural prosthesis [16], [17], mobile humanoid robot [18], and smart home [19]. However, little emphasis is given to the value of BCI-supported collaborative work of people who have a motor disability.

Collaborative brain-computer interface (C-BCI) is defined in this study as a non-invasive electroencephalogram (EEG)-based BCI that supports collaboration in which, by working in a group of two, users can help each other and perform tasks jointly. Depending on the collaboration mode, users may take turns or work simultaneously to perform tasks at their own pace. However, there has been a general lack of understanding of, or inattention to, issues related to BCI-supported collaborative work. For example: Can people with motor disabilities perform a task jointly with other people (with or without motor disability) only through means of their brain activity? How then can BCIs best support their collaborative work? To empirically examine these new research questions, we developed a C-BCI for control – Brainbot, which allows participants to jointly move a ball by controlling a robot arm. It is intrinsic that C-BCIs, compared to individual BCIs, can provide greater benefit to those with motor disabilities. Collaboration can foster the sharing of knowledge, ideas, and skills, and plays an important role in areas of art, academia, business, and scientific research [20], [21]. Research has also indicated that group decisions are superior to individual decisions in many different aspects [22], [23].

More recently, researchers have begun to notice the advantage of C-BCIs and argued that C-BCIs should be investigated to better serve people with motor disabilities. Some initial research has been conducted showing the possibility and potential to develop C-BCIs [13], [24]–[28]. For example, Wang & Jung [27] have examined the performance of C-BCI in a task of movement planning and found that C-BCI setups could result in a better overall performance than a single user BCI system. Eckstein et al. [24] has examined the possibility of integrating the brain activity of group members in the decision making process and found that the combined neural activities could lead to decisions as accurate as the combined behavioral activities in less time. Additionally, Poli and his colleagues [25] investigated C-BCI in a space navigation task and found that C-BCIs produced significantly superior trajectories than single user BCI in a space navigation task. However, it should be noted that all of the aforementioned studies employed able-bodied participants. Also, previous research generally used much simpler tasks and users were not directly controlling any external equipment. Therefore, further research is needed to investigate the collaborative behavior of people with motor disabilities in more complex tasks.

SSVEP-based BCI is chosen for the following reasons: Firstly, SSVEP-based BCI requires almost no initial training, which provides a great advantage over most other BCI systems [29]. Secondly, research has shown that SSVEP-based BCI can provide fast and reliable communication as a non-invasive BCI [30], [31] and also has the advantage of high signal-to-noise ratio (SNR) and robustness to artifacts [32]. Finally, general design solutions exist for SSVEP. For example, LED is preferred to provide visual stimuli and low to medium frequency is generally used [33]. Therefore, SSVEP-based BCI is a good test bed for research relating to C-BCI.

B. Luminance Contrast

Luminance contrast describes the relative brightness of the light source in comparison to another light source and is an extremely important aspect of SSVEP-based BCI design. Research shows that both ambient luminance and contrast can affect the visual acuity of older adults [34], Research shows that the amplitude of the evoked potentials is positively correlated to the logarithm of contrast [35]. Spekreijse [36] found that brightness and modulation depth can increase SSVEP amplitudes. Nam, Li, & Johnson [37] have shown a significant effect for interface color contrast on most P300 measures for able-bodied participants. Participants had significantly higher accuracy, larger information transfer rate (ITR), larger amplitude, and shorter latency in high interface color contrast as compared to low interface color contrast.

Zemon & Gordon [38] investigated the mechanisms of luminance contrast in humans using arrays of isolated-checks and discovered that a raster of dark squares, rather than bright squares, elicit larger responses when
emerging from a neutral background. Bieger & Garcia-Molina [39] investigated the effect of luminance contrast within BCI by testing eight conditions where both foreground and background stimuli luminance was varied. However, the results differed from Zemon & Gordon’s findings [38] in that bright stimuli elicited larger responses over dark stimuli.

C. Motor Disability

Sellers & Donchin [3] have evaluated the effectiveness of a P300-based BCI system with 3 amyotrophic laterals sclerosis (ALS) patients and 3 non-ALS controls and found that non-ALS controls had higher average accuracy than ALS patients. Research by Volosyak et al. [4] evaluated the Bremen SSVEP based BCI with 32 participants including 8 handicapped users in a spelling task. Their result showed that able-bodied participants had significantly higher accuracy than handicapped participants. However, there was no significant difference for ITR. They concluded that there was almost no difference between the disabled and healthy subjects.

Münßinger et al. [40] examined the P300-Brain Painting with 3 ALS patients and 10 healthy participants. Their result showed that healthy participants had higher accuracy and ITR than ALS patients in both copy-spelling and copy-painting. Li and his colleagues [2] have examined the effects of interface type and screen size of a P300 Speller with 10 participants with severe motor disabilities and 10 able-bodied participants as a control group. Their results were consistent with Münßinger et al. [40]. Lim, Hwang, Han, Jung & Im [41] have examined the SSVEP-based BCI with the binary intentions classifications. Their results showed that healthy participants had higher average classification accuracy than the ALS patient.

III. METHODS

A. Participants

Fourteen participants (8 male, 6 female) with motor disabilities (ALS) were recruited from the local ALS association and community, with mean (M) age of 56.4 years old (Standard Deviation, SD = 11.3). Fourteen able-bodied participants of similar ages (8 male and 6 female) were recruited from the local community, with mean (M) age of 52.5 years old (Standard Deviation, SD = 8.5). Participants were rejected if younger than 18 years of age or if suffering or previously suffered from seizures, epilepsy, or skin allergies [42]. All eligible participants were compensated $20/hour for participation and provided with printed informed consent form. The study was approved by institutional review board (IRB) at North Carolina State University (NCSU).

B. Apparatus

The hardware included an amplifier (g.tec Medical Engineering), an EEG cap, electrodes, a g.GAMMAbox (g.tec Medical Engineering), EEG gel, syringes, blunt needles, and LED lights. The frequency detection software and SSVEP-based C-BCI were used, developed by BCI Lab at NCSU.

**LED lights:** Cost effective LED lights were developed as visual stimuli to elicit SSVEP responses. LEDs were highly customized and can be easily adjusted to different colors and frequencies via a programmable Micro Control Unit (Fig. 1).

**Frequency Detection Software:** A frequency detection software was developed to detect the frequency to which the participant is sensitive.

**SSVEP-based C-BCI:** Brainbot is a C-BCI and enables a pair of users to jointly perform a physical control task (Fig. 2). BrainBot consists of a robotic arm, three target locations (station 1 (St1), station 2 (St2) and station 3 (St3)), ten LEDs and a rubber ball. Brainbot is constructed using the LEGO Mindstorms NXT kit communicating with a computer via a Bluetooth medium.
In an example task, participants would be asked to grab (G) a ball, move and release (R) it to one of three target locations (i.e., St1, St2, and St3) alone or collaboratively with their partner by focusing on the LED light corresponding to a desired action/motion. St2 is the neutral position, in which the robot arm and the ball is initially placed for any task.

C. Independent Variables

1) Collaboration Mode

Three types of collaboration modes were used in this study. In each collaboration mode, participants are provided a predetermined sequence of “correct” movements. Any visual stimulus that has been chosen three times in a row constitutes a successful movement to be carried out by the robot. However, only movements following the predefined path constituted correct moves. For example, if the predefined path was G → St3 → R, and the participant(s) performed the following six movements: G → St1 → St3 → St2 → St3 → R, then only three of the sequential movements are considered “correct” (i.e., G, the first instance of St3, and R). Both the participants and the experimenters know the predefined path. The BCI system will send any successful movement to the robot. The three collaboration modes are:

(a) Individual Mode (no collaboration): Each participant (1 and 2) performed the task individually. When a successful but incorrect movement (not the same as the predefined path) was made, the participant must correct it before making a subsequent movement in the predetermined path.

(b) Sequential Mode: Two participants (1 and 2) took turns performing the task as a group. When an incorrect movement was made, whoever made it must correct the movement before making a subsequent movement in the predetermined path. For two participants (1 and 2) to perform a 6-sequence control, the pattern would look like: 1-2-1-1-2-1-2-1, where 1 and 2 represent participant 1 and participant 2, respectively. When one participant is working, the other participant is at rest.

(c) Simultaneous Mode: Two participants performed the task together (simultaneously) as a group. Each participant’s brain signal was processed separately at first, and then combined together following the defined rule. There are three cases: (I) If there is no successful classification, no command will be sent to the robot; (II) If there is only one successful classification (either 1 or 2), the corresponding command will be sent to the robot; (III) If there are two successful classifications (i.e., both 1 and 2), then the average spectral power will be chosen and the corresponding command will be sent to the robot.

If no successful movement is made or a successful but incorrect one (not the same as the predefined path) is made, both participants need work together to make the correct movement again. For two participants (e.g., 1 and 2) to perform a 6-sequence control, the pattern could be pretty random. For example, if participant 1 performed the first 3 sequences and participant 2 performed the second 3 sequences, the pattern would look like 1-1-1-2-2-2-2.

2) Luminance Contrast

Web contrast [43] was used to define luminance contrast (\(C_L\)):

\[
C_L = \frac{L_v - L_s}{L_s}
\]

where \(L_v\) is the luminance of the visual stimulus and \(L_s\) is the luminance of the background. It is obvious to see that when the background is more luminous (i.e., brighter) than the visual stimulus, then \(-1 < C_L < 0\). Likewise, when the background is darker than the visual stimulus, \(C_L > 0\).

Luminance contrast was controlled by adjusting the background luminance. There were only two light sources in the lab: the visual stimuli (LED lights) and the room’s overhead lights. Both light sources maintained a stable luminance. Two levels of background luminance were controlled: low (light off, 55 cd/m\(^2\)), and high (light on, 750 cd/m\(^2\)). The LED lights have luminance of 150 cd/m\(^2\). Correspondingly, there were two levels of luminance contrast: 1.73 and –0.8.

3) Motor Disability

The participants were classified into two equal-sized groups: able-bodied participants and participants with motor disabilities (ALS participants).

D. Dependent Variables

1) Task Performance

Accuracy (%): Accuracy is the ratio of the number of correct robot movements over the total number of robot movements. For example, if the predefined path was G → St3 → R and the participant performed six movements G → St1 → St3 → St2 → St3 → R, then the accuracy would be 50% (3/6).

Information Transfer Rate (ITR) (bits/minute): ITR has been widely used in BCI research [45] and conveys how many bits can be transferred per minute. It can be calculated by two steps. At first, the number of bits transferred per trial (B) would be calculated. Then, ITR can be obtained by dividing B by the duration of the trial. The following formula, defined by Pierce [44] and Wolpaw et al. [45], would be used to calculate the number of bits transmitted per trial:

\[
B = \log_2 N + A \log_2 A + (1 - A) \log_2 \left( \frac{1 - A}{N - 1} \right)
\]

\[
ITR = \frac{B}{T}
\]
where \( N \) is the number of possible targets (number of visual stimuli in this study), \( A \) is the probability that the target was accurately classified (accuracy here), and \( T \) is the duration of the trial in minutes.

Task Completion Time (seconds): Task completion time measures the time duration needed to perform one trial.

2) Brain Activity

Spectral Power (\( \text{mV}^2 \)): Spectral Power is defined as,

\[
P = \lim_{T \to \infty} \frac{1}{2T} \int_{-T}^{T} x(t)^2 \, dt
\]

where \( x(t) \) is the amplitude of the EEG signal and \( T \) is the duration of one cycle [46].

3) User Evaluations

User Preference: Participants were asked to choose their favorite experiment condition out of 6 conditions (3 collaboration modes \( \times \) 2 luminance contrasts) when finished after the experiment.

E. Data Acquisition and Signal Processing

1) Data Acquisition

SSVEP brain waves were recorded using an EEG cap embedded with 16 channels (Fz, C3, C2, C1, CP3, P7, P3, Pz, P4, POz, PO2, O1, O2, and O2) based on the modified 10-20 system of the International Federation [47]. Brain signals from channel O1 and O2 were used as control signal. Figure 3 showed the electrode montage used in the study. Recordings were referenced to the left mastoid and grounded to location AFz [42].

SSVEP brain waves were amplified with a g.USBamp amplifier (g.tec Medical Engineering). Data collection, online signal processing and offline data analysis were conducted using LabVIEW-based software developed by BCI Lab at North Carolina State University.

2) Signal Processing

Many methods have been used in the SSVEP signal classification, such as Harmonic Sum Decision (HSD) [16],[48], Canonical Correlation Analysis (CCA) [30], Linear Discriminant Analysis [49], and Minimum Energy Combination [31]. HSD was chosen in the current study for its efficiency and simplicity. Data processing included two phases: preprocessing and classification. In the preprocessing phase, raw EEG data was filtered and transformed into the frequency domain. In the classification phase, the brain signal was classified and

![Figure 3: Electrode Montage.](image-url)
a corresponding decision was sent to the NXT robot. EEG raw data was sampled with a rate of 512 Hz. It was filtered by a 0.1 Hz high pass filter and a 75 Hz low pass filter, notch filtered by 60 Hz. Then, the acquired time domain EEG data was transformed into frequency domain to get SSVEP response in the power spectrum by FFT. The time window was 3 seconds and the window sliding speed was 0.25 second. Next, the EEG data in frequency domain was adjusted by the baseline. After the process of normalization and averaging, the relative spectral data were sent to a classifier, in which the frequency with the maximum average spectral power would be chosen preliminarily. If a frequency was chosen 3 times in a row, it would result in a corresponding movement command, which would be sent to the robot to carry out. Figure 4 displayed the flow of the signal processing.

F. Experimental Task
In the study, each participant must perform a predetermined sequence of six movements (G → St3 → R → G → St1 → R) in each condition either alone (individual mode) or together with the teammate (sequential or simultaneous mode). In other words, the participant must grab (G) the rubber ball in the neutral position (St2), then move to St3 and release (R) the ball, then grab (G) the ball and move to St1, then release (R) the ball.

Four movements were available in each task: G, R, St1 and St3. The four identified user-specific visual stimuli were used for each participant to optimize movement selection accuracy. The four visual stimuli were set up in front of the participant and apart from each other to prevent the interference based on the pilot test. The same layout of the visual stimuli was applied to each participant based on the pilot test. There were 6 conditions (3 collaboration modes × 2 luminance contrasts) for each participant with 1 trial in each condition. The 8 trials (4 trials in individual mode + 2 trials in sequential mode + 2 trials in simultaneous mode) were completely randomized in presentation to reduce any trial order effect. A 3-minute break was provided between each trial.

G. Procedure
All experiments were conducted in a quiet room. Before the experiment, participants were instructed on the basic procedure of the experiment and how to respond to visual stimuli. Participants were provided a comfortable chair and required to avoid any unnecessary gross body movement and to focus on the corresponding visual stimuli at each movement in a relaxed manner during the experiment. No interferences or hints were provided during the experiment.

At first, the user-specific visual stimuli (frequency & color) was determined for each participant based on spectral power values using the frequency detection software. During the process, the participant was asked to focus on LED lights with different frequencies and colors. The four LED lights with different frequencies which produced the biggest spectral power were chosen as the visual stimuli to control the robot.

Then, they performed the robot control task for 8 trials. Once the system is set up, participants were asked to close their eyes for 2 minutes for the system to record the baseline of EEG. Next, the participants were asked to open their eyes and perform the first task. When finished, a three-minute rest was given. Then, participants began the second task. This process was repeated until all 8 trials were completed. Next, each participant was required to complete a single questionnaire to evaluate the system and experimental condition. Finally, participants were compensated by cash and debriefed with the help of the experimenters. The whole procedure lasted approximately 1 hour.

H. Data Analysis
The study was a balanced 2 × 3 × 2 mixed design. There were two within-subjects factors (Collaboration Mode and Luminance Contrast), and one between-subjects factor (Motor Disability).

IV. RESULTS
Table 1 summarized the significant effects for performance measures.

<table>
<thead>
<tr>
<th>PERFORMANCE MEASURE</th>
<th>EFFECT</th>
<th>F-VALUE</th>
<th>P-VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCURACY</td>
<td>COLLABORATION MODE</td>
<td>F_{2,26} = 36.51</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td></td>
<td>LUMINANCE CONTRAST</td>
<td>F_{1,12} = 72.14</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>COMPLETION TIME</td>
<td>COLLABORATION MODE</td>
<td>F_{2,26} = 43.00</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td></td>
<td>LUMINANCE CONTRAST</td>
<td>F_{1,12} = 73.18</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>ITR</td>
<td>COLLABORATION MODE</td>
<td>F_{2,26} = 56.45</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td></td>
<td>LUMINANCE CONTRAST</td>
<td>F_{1,12} = 182.59</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>SPECTRAL POWER</td>
<td>COLLABORATION MODE</td>
<td>F_{2,26} = 8.70</td>
<td>0.0014</td>
</tr>
<tr>
<td></td>
<td>LUMINANCE CONTRAST</td>
<td>F_{1,12} = 53.12</td>
<td>&lt;0.0001</td>
</tr>
</tbody>
</table>

FIGURE 4 Signal Processing Diagram.
The completion time decreased by 39.4% from individual mode to simultaneous mode, and decreased by 28.8% from low luminance contrast to high luminance contrast.

A. Task Performance

1) Accuracy (%)
The analysis revealed a significant main effect of collaboration mode \((F_{2,24} = 36.31, p < 0.0001)\) and luminance contrast \((F_{1,12} = 72.14, p < 0.0001)\). Tukey’s HSD test indicated that accuracy in simultaneous mode \((M = 0.868, SD = 0.126)\) was significantly higher than that in sequential mode \((M = 0.728, SD = 0.115)\) \((F_{1,2} = 38.43, p < 0.0001)\) and individual mode \((M = 0.684, SD = 0.129)\) \((F_{1,2} = 66.63, p < 0.0001)\). However, accuracy was not significantly different between individual mode and sequential mode \((F_{1,12} = 3.85, p = 0.063)\), although the mean accuracy in sequential mode was higher than that in individual mode. Accuracy in high luminance contrast \((M = 0.839, SD = 0.126)\) was significantly higher in comparison to low luminance contrast \((M = 0.681, SD = 0.112)\). Results revealed no significant effect of motor disability \((F_{1,12} = 0.01, p = 0.9274)\). Accuracy for ALS participants \((M = 0.762, SD = 0.147)\) and able-bodied participants \((M = 0.758, SD = 0.139)\) was not significantly different. No significant interaction effects were found between collaboration mode and motor disability \((F_{2,24} = 0.91, p = 0.4151)\), luminance contrast and motor disability \((F_{1,2} = 0.23, p = 0.6405)\), or collaboration mode and luminance contrast \((F_{2,24} = 1.65, p = 0.2125)\).

2) Completion Time (seconds)
The analysis revealed a significant main effect of collaboration mode \((F_{2,24} = 43.00, p < 0.0001)\) as well as a significant effect of luminance contrast \((F_{1,12} = 73.18, p < 0.0001)\). Tukey’s HSD test indicated that completion time in simultaneous mode \((M = 30.95, SD = 10.33)\) was significantly shorter than that in sequential mode \((M = 45.19, SD = 12.23)\) \((F_{1,12} = 40.62, p < 0.0001)\), which was also significantly shorter than that in individual mode \((M = 51.10, SD = 11.63)\) \((F_{1,2} = 7.01, p = 0.0141)\). Meanwhile, completion time in high luminance contrast \((M = 35.29, SD = 10.32)\) was significantly shorter than that in low luminance contrast \((M = 49.53, SD = 13.93)\). The completion time also indicated the clinical difference. For example, the completion time decreased by 39.4% from individual mode to simultaneous mode, and decreased by 28.8% from low luminance contrast to high luminance contrast. However, the analysis revealed no significant effect of motor disability \((F_{1,12} = 0.48, p = 0.5005)\). Completion time of ALS participants \((M = 43.55, SD = 13.08)\) was not significantly different from able-bodied participants \((M = 41.28, SD = 15.18)\). No significant interaction effects were found between collaboration mode and motor disability \((F_{2,24} = 3.12, p = 0.0622)\), luminance contrast and motor disability \((F_{1,2} = 0.66, p = 0.4316)\), or collaboration mode and luminance contrast \((F_{2,24} = 4.89, p = 0.0719)\).

3) Information Transfer Rate (bits/minute)
The ANOVA revealed a significant main effect of collaboration mode \((F_{2,24} = 56.45, p < 0.0001)\) as well as a significant effect of luminance contrast \((F_{1,12} = 182.59, p < 0.0001)\). Tukey’s HSD test indicated that ITR in simultaneous mode \((M = 3.16, SD = 2.18)\) was significantly higher than that in sequential mode \((M = 1.24, SD = 1.04)\) \((F_{1,12} = 13.62, p < 0.0001)\), which was significantly higher than that in individual mode \((M = 0.87, SD = 0.60)\) \((F_{1,2} = 6.01, p = 0.0219)\). ITR in simultaneous mode was also significantly higher than that in individual mode \((F_{1,12} = 59.72, p < 0.0001)\). Meanwhile, ITR in high luminance contrast \(M = 2.54, SD = 1.97\) was significantly higher than that in low luminance contrast \((M = 0.97, SD = 0.99)\). However, the analysis revealed no significant effect of motor disability \((F_{1,12} = 0.05, p = 0.8209)\). No significant interaction effects were found between collaboration mode and motor disability \((F_{2,24} = 1.54, p = 0.2345)\), luminance contrast and motor disability \((F_{1,12} = 0.15, p = 0.7084)\), and collaboration mode and luminance contrast \((F_{2,24} = 0.58, p = 0.5660)\).

B. Brain Activity

The analysis revealed a significant main effect of collaboration mode \((F_{2,24} = 8.70, p < 0.0014)\) as well as a significant effect of luminance contrast \((F_{1,12} = 53.12, p < 0.0001)\). Tukey’s HSD test indicated that spectral power in individual mode \((M = 0.787, SD = 0.269)\) was significantly smaller than that in sequential mode \((M = 1.049, SD = 0.407)\) \((F_{1,12} = 16.48, p = 0.0005)\) and simultaneous mode \((M = 0.972, SD = 0.273)\) \((F_{1,12} = 8.15, p = 0.0087)\). However, spectral power was not significantly different between sequential mode and simultaneous mode \((F_{1,12} = 1.45, p = 0.2399)\), although the mean spectral power in sequential mode was higher than that in simultaneous mode. Meanwhile, spectral power in high luminance contrast \((M = 1.028, SD = 0.351)\) was significantly bigger than that in low luminance contrast \((M = 0.844, SD = 0.300)\). However, the analysis revealed no significant effect of motor disability \((F_{1,12} = 0.01, p = 0.9358)\). Spectral power of ALS \(M = 0.942, SD = 0.359\) and able-bodied participants \((M = 0.930, SD = 0.320)\) was not significantly different. No interaction effects were found between collaboration mode and motor disability \((F_{2,24} = 0.44, p = 0.6481)\), luminance contrast and motor disability \((F_{1,12} = 0.22, p = 0.6489)\), or collaboration mode and luminance contrast \((F_{2,24} = 0.48, p = 0.6256)\).

C. User Evaluations

A two-way contingency analysis was conducted for user evaluations at first. Fisher’s exact Fisher Test was utilized.
because some cells had less than 5 observations [50]. If there was no association between the two factors, a one-way frequency analysis was conducted.

Table 2 summarizes ALS participants’ user preference of experiment conditions.

The Fisher Exact Test showed that there is no association between collaboration mode and luminance contrast (p = 0.3407). A one-way frequency analysis of collaboration mode revealed no significant difference in user preference (X² = 4, p = 0.1353). A one-way frequency analysis of luminance contrast revealed significant difference in user preference (X² = 7.1429, p = 0.0075) showing that ALS participants preferred high luminance contrast over low luminance contrast.

Table 3 summarizes able-bodied participants’ user preference of experiment conditions.

The Fisher Exact Test showed no association between collaboration mode and luminance contrast (p = 0.5385). A one-way frequency analysis of collaboration mode revealed no significant difference in user preference (X² = 0.2587, p = 0.5930). A one-way frequency analysis of luminance contrast revealed significant difference in user preference (X² = 4.5714, p = 0.0325) showing that able-bodied participants preferred high luminance contrast over low luminance contrast.

V. DISCUSSION

A. Effects of Collaboration Mode

Results showed that collaboration mode had a significant main effect on both task performance and brain activity. Sequential mode can produce significantly better task performance than individual mode. Sequential mode also produced a significantly better task performance than individual mode with the exception of accuracy (there was no significant difference).

Sequential Mode vs. Individual Mode: The results of this research are similar to those from Poli et al. [25], [13], Eckstein et al. [24], and Wang & Jung [27]. Research of group vs. individual showed that groups perform better than the average individual [51], [52], [53]. The result of the study was also consistent with this viewpoint.

Simultaneous mode provides an advantage over individual mode in SSVEP-based BCI in the form of group work efficiency. The error cancellation property of group work makes the simultaneous mode much more efficient than the individual mode. For example, in individual mode, the individual user had to make each movement alone and correct errors alone. In simultaneous mode, however, correct movement could be made if chosen by either participant. Meanwhile, the error could be corrected by either user. This mechanism greatly improved the accuracy as well as ITR and reduced the task completion time. Social facilitation could enhance brain activity in simultaneous mode tasks compared to individual mode tasks [54], [55]. Likewise, social facilitation could also contribute to greater task performance in simultaneous mode compared to individual mode. The presence of a teammate may cause each participant to have higher arousal level and enhanced focus to the visual stimulus [54], which in turn helped to produce stronger brain activity and greater task performance [56].

Sequential Mode vs. Individual Mode: Sequential mode is actually a modification of individual mode. The only difference is that two participants take turns completing the task, even though they can’t help each other select movement or correct errors. In other words, each individual would perform only three movements rather than six, providing more opportunities for rest after each correct movement.

This is a critical advantage for sequential collaboration - less burden and more downtime - which could reduce the participant’s tiredness, elevate the alertness, potentially produce better task performance (shorter completion time and bigger ITR) and stronger brain activity. Although the accuracy was not significantly higher in sequential mode (the average accuracy was higher), it is reasonable to believe that the advantage should be more significant as the number of movements increases in the task. As discussed earlier, social facilitation should also contribute to the advantage of sequential mode.

User preference also supports the popularity of the collaborative mode. Results showed that about 71.4% ALS participants and 100% able-bodied participants preferred collaborative mode. Participants commented that they “enjoyed the collaboration process with the teammate in the collaborative mode” and “were more efficient when controlling the robot together with the teammates”, because they could get help from the teammate if they make a wrong move. However, some ALS participants expressed their concern about the collaborative mode. One ALS participant commented that she preferred...
The error cancellation property of group work makes the simultaneous mode much more efficient than the individual mode.

the individual mode because she “was always in control of the process”. Another ALS participant said he “couldn’t believe in his teammate and would rather perform the task alone”. These comments surfaced some very important research question: How can C-BCI system provide users with motor disabilities a greater sense of control? What training paradigm could bolster mutual trust? The result proves that the collaborative BCI is workable and more efficient than individual mode. It indicates the feasibility of developing more collaborative applications for those with motor disabilities in order to improve their quality of life.

B. Effects of Luminance Contrast
Results showed that luminance contrast had a significant main effect on task performance and brain activity. Participants had a significantly higher accuracy, a shorter completion time, a bigger ITR and a bigger spectral power in high luminance contrast compared to low luminance contrast. Nam et al. [37] and Li, Bahn, Nam & Lee [57] have obtained similar results in a P300-based BCI.

One explanation is that participants could perceive more intense stimuli in high luminance contrast when all other environmental factors remained unchanged. High luminance contrast could result in brighter, more noticeable visual stimuli compared to low luminance contrast, even when the visual stimuli were the same luminance. From visual sensory theory, rods will function more in high luminance contrast and make green and blue LED lights appear brighter [58]. Since more green and blue were used than red, high luminance contrast produced brighter visual stimuli. As revealed by Campbell & Maffei [35], the amplitude of evoked potentials was linearly increasing as the logarithm of the contrast of the stimulus. The spectral power in high luminance contrast was bigger than that in low luminance contrast. Meanwhile, participants could stay more alert and have better focus in high luminance contrast. Li et al. [57] showed that participants can lose focus when using BCIs. Higher luminance contrast may help participants to remain more focused so as to produce better task performance.

User preference also supports the popularity of the high luminance contrast. Results showed that significantly more participants preferred high luminance contrast. About 85.7% ALS participants and 78.6% able-bodied participants preferred high luminance contrast. Most participants commented that they could “catch” the stimuli more easily in high luminance contrast condition. Some participants said that it was easier to focus on the visual stimulus in high luminance contrast, while it was difficult in low luminance contrast because visual stimuli did not provide enough intensity and required too much concentration. On the other hand, some participants disliked high luminance contrast. One participant said that in high luminance contrast the visual stimuli were too bright while low luminance contrast was more user-friendly and allowed for longer durations of use. Another participant commented that high luminance contrast produced some sort of “burn” effect in the eyes that worsened with time. These comments advanced the following research questions: What level of luminance contrast is most user-friendly for long-term use? Is there a luminance contrast threshold? Further research is needed to answer those questions.

C. Effects of Motor Disability
It revealed no significant main effect of motor disability nor any interaction effect between motor disability and other factors. ALS participants exhibited comparable task performance to the able-bodied participants in all three modes, which differs from the previous research in [2], [3], [40], [41]. Li et al. [2] and Volosyak et al. [4] found that able-bodied participants had better task performance than participants with severe motor disabilities.

In an SSVEP-based BCI, participants need only focus on the visual stimulus, which brings into question whether ALS participants’ motor disabilities could affect the quality and duration of focus. Li et al. [2] argued that participants with motor disability become easily fatigued in shorter duration of focus, which affected their performance in comparison to able-bodied participants. Piccione et al. [59] also found that participants with motor disabilities were unable to avoid head and/or eye movement, which are known to produce artifacts, disturb classification and harm performance.

However, the hypothesized “fatigue” from Li et al. [57] was not observed in this study. No ALS participants requested extra rest during the experiment, indicating that their motor disability didn’t result in extra fatigue. Also, the hypothesized “head or eye movement” by Piccione et al. [59] was not observed in the experiment. ALS participants not utilizing wheelchairs (6 in the study) retained most motor abilities and could control head or eye movement as good as any able-bodied participant. ALS participants utilizing wheelchairs (8 in the study) were unable to move and rested their heads against the back of the wheelchair. In this sense, it is possible to argue that SSVEP-based BCI usage is unaffected by motor disability. Most BCI researchers also hold this paradigm [1], [3], [4], although this postulation has not been significantly supported with a large sample size. Therefore, the empirical data in this study should provide greater substantial support in favor of such a paradigm.

Results of this study indicated that ALS patients could use SSVEP-based BCI as accurately and efficiently as able-bodied people. It upholds the potential of BCI as an alternative method of communication and control for those with motor disabilities in communication and control. It also sheds light on the future research and development of BCI. However, it should be noted that this study didn’t consider the different
stages of ALS patients, which may affect their performance, as shown by McCane and the colleagues [60].

**D. Effects of Collaboration Mode on Individual Performance**

Results showed that participants had better task performance in collaborative mode than individual mode. Then the question is: Is it possible that collaboration elevates individual performance?

To explore this question, accuracy, average individual completion time and spectral power were retrieved from the simultaneous mode and sequential mode. The data was segregated by each individual participant’s movement selections within the task. For example, if participant 1 performed four movements in one trial in simultaneous mode, then, participant 1’s individual data in this trial would be based on those four movements. Since each individual participant may perform different number of movements in simultaneous mode, the individual task completion time would not be comparable. For comparison, average completion time was used in the analysis. ANOVA was conducted to evaluate the effects of collaboration mode on the individual performance.

1) **Accuracy (%)**

The analysis revealed a significant main effect of collaboration mode \((F_{2,32} = 26.23, p < 0.0001)\). Tukey’s HSD test indicated that the accuracy was significantly different between individual mode and simultaneous mode \((F_{1,26} = 52.20, p < 0.0001)\), between individual mode and sequential mode \((F_{1,26} = 10.01, p = 0.0026)\), and between simultaneous mode and sequential mode \((F_{1,26} = 16.50, p = 0.0002)\). The individual accuracy in simultaneous mode \((M = 0.900, SD = 0.157)\) was significantly higher than that in sequential mode \((M = 0.779, SD = 0.207)\), which was also significantly higher than that in individual mode \((M = 0.684, SD = 0.147)\).

2) **Average Completion Time (seconds)**

The analysis showed a significant main effect of collaboration mode \((F_{2,32} = 35.34, p < 0.0001)\). Tukey’s HSD test indicated that the average completion time was significantly different between individual mode and simultaneous mode \((F_{1,26} = 70.06, p < 0.0001)\), between individual mode and sequential mode \((F_{1,26} = 19.84, p < 0.0001)\), and between simultaneous mode and sequential mode \((F_{1,26} = 15.59, p = 0.0002)\). The individual average completion time in simultaneous mode \((M = 3.75, SD = 2.70)\) was significantly shorter than that in sequential mode \((M = 5.99, SD = 4.00)\), which was also significantly shorter than that in individual mode \((M = 8.52, SD = 2.35)\).

3) **Spectral Power (** \(\text{mv}^2\)**

The result revealed a significant main effect of collaboration mode \((F_{2,32} = 4.59, p = 0.0146)\). Tukey’s HSD test indicated that the individual spectral power in individual mode \((M = 0.808, SD = 0.387)\) was significantly smaller than that in sequential mode \((M = 1.049, SD = 0.648)\) \((F_{1,26} = 8.65, p = 0.0049)\) and simultaneous mode \((M = 0.981, SD = 0.547)\) \((F_{1,26} = 4.43, p = 0.0402)\). However, the individual spectral power was not significantly different between simultaneous mode and sequential mode \((F_{1,26} = 0.70, p = 0.4069)\).

Result revealed a significant main effect of collaboration mode for task performance and brain activity. It showed that collaboration not only improved the group performance, but also elevated the individual performance within the group. In other words, participants had significantly better performance collaborating as a group member rather than completing a task alone.

**E. Other Issues**

It should be noted that the ITR reported in this study is not as high as other studies. One reason is that we want to test the feasibility and sustainability of SSVEP-based collaborative BCI. To make sure that the system is sustainable for future use in real life, three successful selections in a row is needed to make a robot movement or make a correction. These greatly increased the task completion time. We elevate the difficulty level of the task on purpose. Actually, most participants can use one successful selection to make a robot movement or make a correction. In the current system, the participant will take at least 200% more time as it is actually needed to focus on the visual stimuli and produce the command to move the robot. Meanwhile, the robot is a very simple LEGO demo. The Grab/Release movement takes around 1 second each and other robot movement around the 3 stations takes around 2 second each. During that time, the BCI system will not work in order to guarantee that the robot will not receive another command during the movement. On the whole, the robot movement will take around 40~45% of the task completion time. We believe that more advanced robots will integrate with the BCI system more efficiently and greatly reduce the task time, and therefore increase the ITR.

**VI. CONCLUSION**

This study investigated the effects of collaboration mode, luminance contrast and motor disability on task performance, brain activity and user evaluations. Results revealed significant main effects of collaboration mode and luminance contrast, while no effect of motor disability. These results could provide precious empirical data and invaluable insights to the real-world applicability of the SSVEP-based BCI applications for people with motor disabilities. Most importantly, this research demonstrated that people with motor disabilities can use C-BCI as efficient as able-bodied people, which proved the potential of C-BCI to help those with motor disabilities. Future research should focus on the layout of the visual stimuli when there are more targets so as to prevent the interference between each other. Different stage of the ALS patients should also be considered in order to customize the applications for all potential users. Meanwhile, the collaboration in this study uses a method of combination based on each participant’s selection. Future research may attempt combining all the participants’ brain
Chang-Shing Lee and Mei-Hui Wang  
Department of Computer Science and Information Engineering, National University of Tainan, TAIWAN

Shi-Jim Yen  
Department of Computer Science and Information Engineering, National Dong Hwa University, TAIWAN

Ting-Han Wei and I-Chen Wu  
Department of Computer Science, National Chiao Tung University, TAIWAN

Ping-Chiang Chou and Chun-Hsun Chou  
Taiwan Go Association, TAIWAN

Ming-Wan Wang  
Nihon Ki-in Go Institute, JAPAN

Tai-Hsiung Yang  
Haifong Weiqi Academy, TAIWAN

Human vs. Computer Go: Review and Prospect

Abstract
The Google DeepMind challenge match in March 2016 was a historic achievement for computer Go development. This article discusses the development of computational intelligence (CI) and its relative strength in comparison with human intelligence for the game of Go. We first summarize the milestones achieved for computer Go from 1998 to 2016. Then, the computer Go programs that have participated in previous IEEE CIS competitions as well as methods and techniques used in AlphaGo are briefly introduced. Commentaries from three high-level professional Go players on the five AlphaGo versus Lee Sedol games are also included. We conclude that AlphaGo beating Lee Sedol is a huge achievement in artificial intelligence (AI) based largely on CI methods. In the future, powerful computer Go programs such as AlphaGo are expected to be instrumental in promoting Go education and AI real-world applications.

I. Computer Go Competitions
The IEEE Computational Intelligence Society (CIS) has funded human vs. computer Go competitions in IEEE CIS flagship conferences since 2009. Fig. 1 shows the year and the location of the conferences. The descriptions of competitions held from 1998 to 2016 are listed in detail in an online version of this article [1]–[8]. The handicaps for the human vs. computer 19×19 game have been decreased from 29 in 1998 to 0 in 2016. The handicaps for the human vs. computer 9×9 game have been decreased from 29 in 1998 to 0 in 2016. The skill difference between professional ranks is much less than one stone for every rank difference. Go is typically played on 19×19 size boards, but 9×9 size boards are also common for beginners. The complexity of the 9×9 game is far less than the standard game, and the 9×9 game had been one of the interim goals for computer Go programs. Go is a game that is inherently biased for the first player to play, Black. To compensate for this first player advantage, White is awarded additional points at the end of the game, which is referred to as komi. The related statistics for the IEEE CIS human vs. computer Go competitions are listed in the online version of this article [8]. It is worth noting that with a komi of 7.5, White may end up with an advantage in both 9×9 games and handicapped 19×19 games, regardless of whether White is played by humans or computers. Fig. 2 shows the certificate awarded to the MoGoTW program (16 cores/48 GB/
9 × 9) by the Taiwan Go Association in 2010 for playing at a 3D amateur level [1].

In this article, we attempt to demonstrate the massive barrier that competing programs need to overcome to achieve comparable performance to AlphaGo. For more information, eleven programs (Aya, CGI, Coldmilk/Jimmy, Erica, Fuego, MoGo/MoGoTW, Many Faces of Go, Pachi, and Zen) from 7 countries that have participated in past IEEE CIS conferences are listed in alphabetical order in the online version of this article [8].

II. AlphaGo

In this section, we briefly introduce the past techniques used in computer Go programs, then provide an estimate for why AlphaGo is able to outperform contemporary programs so dramatically. Currently, Monte Carlo tree search (MCTS), minorization–maximization (MM), and deep convolutional neural networks (DCNNs) have demonstrated great success in Go. MCTS was successfully applied to Go in 2006 [9], [10], leading to a significant improvement in playing skill. One year later, MM was applied so that programs may recognize move patterns using supervised learning, with expert game records as the training sample [11]. Though not as revolutionary as MCTS, MM has also had a long-lasting impact on Go programs from 2007 up to 2014.

In December of 2014, two teams (one of which is the DeepMind AlphaGo team) applied DCNNs to Go independently [12], [13]. Clark and Storkey [12] first published a paper that applied DCNNs to Go, which, when given a game position, could estimate how expert human players respond with a prediction rate of 41%-44%, exceeding the rate of previous methods. Meanwhile, DeepMind's method, which was released 10 days later, had a prediction rate of 55%. Among many of DCNN's applications, it has seen success in image and video recognition. When applied to Go, DCNN is able to recognize move patterns at a significantly lower error rate than MM. For this reason, most state-of-the-art computer Go programs use MCTS combined with either MM or DCNN.
AlphaGo is able to perform leaps and bounds above other contemporary programs because of its extensive use of high quality neural networks, which cannot be easily reproduced by other teams due to insufficient experience and/or inadequate hardware resources. To illustrate, let us consider the three main neural networks used in AlphaGo: a supervised learning (SL) policy network, a reinforcement learning (RL) policy network, and the value network [14]. Both the SL policy network and the value network were used in AlphaGo during competitive play; the RL policy network was used only for generating the training samples for the value network. The SL policy network takes a game board position and attempts to guess where expert players will play next. This SL process was performed with 30 million game positions, and involved 340 million training steps, taking a total of three weeks with 50 graphic processing units (GPUs) [14]. The SL algorithm tends to mimic what it has learned from game records instead of favoring moves that yield the highest winning rate when given a choice. To improve on this, the SL policy network was used to train a separate policy network using RL. Training for the RL policy network takes one day with 50 GPUs. The key to AlphaGo’s playing skill is its

FIGURE 3 Game record (Black is Lee Sedol, White is AlphaGo, and White won by resignation) for match No. 5 on Mar. 15, 2016.
value network, which was trained through RL with 30 million self-play game positions. The training process takes about one week with 50 GPUs, for a total training time of four weeks and a day for all three networks.

The most time-consuming and most difficult process to reproduce, however, is not the training of these three networks, but the generation of self-play game positions. For each of the 30 million self-play positions, 100 playouts are performed; for each playout, we assume on average 200 moves until game completion, so a total of 600 billion move data samples need to be generated to train the value network. Let us assume, for the sake of demonstration, that a research team has access to four GPUs. The training of the three networks will take \([4 \times 7 \times 50 \div 200 \times 2411 \div 362.5 \approx 208\text{ million seconds, which}

\[\text{is equivalent to}\]

\[
\begin{align*}
7 \text{ days} \times 4 \text{ GPUs} & = 28 \text{ days} \\
\text{Total processing time} & = 7 \text{ days} + 28 \text{ days} = 35 \text{ days}
\end{align*}
\]

To achieve this, a total of 35 days is required. In addition to the total time required for training, the distributed resources does not even take into account the storage of the generated data samples and the resources needed for the generate of expert game records that are used to train the initial SL policy network, etc. This quick estimate of required resources does not even take into account the knowledge and experience that the DeepMind team has acquired since its inception. As a side note, the distributed version of AlphaGo uses 280 GPUs [14].

### III. Human Intelligence View

Demis Hassabis, CEO and Co-Founder of Google DeepMind, described Go as the “Mt. Everest” of AI [15] because Go is a very complex board game that requires intuitive, creative, and strategic thinking [16]. In the past decade, the techniques of MCTS had revolutionized the field of computer game-playing. The playing strength of computer Go programs has progressed to about a four-stone handicap against top professional Go players in 2012-2015. More precisely, until AlphaGo’s emergence in Oct. 2015, the world’s strongest program, Zen, was able to beat a top professional Go player with a handicap of four stones, while losing when the handicap was decreased to three stones. Google DeepMind introduced a new approach that combined MCTS with deep learning in their program AlphaGo [14], which subsequently broke this four-stone handicap barrier in the recent competition with Lee Sedol (9P) in Korea, in Mar. 2016. AlphaGo’s performance sent shock waves through the community of professional Go players and AI researchers. Since then, people on the Internet and media, particularly in Go-playing cultures such as in Korea, Taiwan, Japan, and China, were buzzing with discussion on related topics [17]–[19]. In this section, we invited three high-level professional Go players who have spent time helping the development of computer Go programs, including Coldmilk and MrGoTW, and who have also been part of shaping Go trends for many years, to comment on the game results of the challenge match from Mar. 9 to Mar. 15, 2016. Fig. 3 shows the game record for match No. 5 and Comment 1 lists three professional Go players’ brief commentaries. For the other four matches’ commentaries and the full commentary on match No. 5, readers can refer to the online version of this article [8]. Additionally, readers may find details for the Go terminologies used in Comments 1 and 2 at Sensei’s Library [20]. From the perspective of professional Go player (Ping-Chiang Chou/6P), the strengths and weaknesses of AlphaGo are listed in Comment 2.

### IV. Discussions and Future Studies

It has been estimated that when human players play against a new computer program for the first time, their strengths are usually weakened by about 1-2 levels. This
is because human players will usually play a probe move when they have no idea about the program's strength. Another key point is that Go programs tend to be designed differently from how human Go players tend to think; winning ten points is essentially the same as winning one point as far as the program's objective function is concerned. As a result, the Go programs tend to maximize the winning rates, while human players tend to maximize territory. Additionally, programs have other advantages when playing against humans, such as relatively strong raw computational power and the lack of facial expressions, all of which in contrast are human disadvantages. There are weaknesses in computer programs, however, such as hidden bugs, imperfect judgement on the overall situation at the beginning of the game, and imperfect parameter adjustment. According to cognitive psychologists [21], the difference between experts (for example, Go teachers or professional Go players) and newcomers is that the former can store a variety of Go patterns in their brains, allowing them to quickly find a more precise move than the latter. Upon seeing the formed shapes of stones on the board, they are able to scan their memory to find the best matching pattern, then play an effective move. Newcomers, on the other hand, are short of such a quick connection.

Computer Go, in particular AlphaGo, has been an extremely hot topic lately, not only to AI researchers but also to the general public. The purpose of this article is to help the readership better understand how the development of computer Go programs has arrived at this milestone of winning against one of the top human players, and how CIS has been involved in this process. This article's main contribution is to place the win of AlphaGo over Lee Sedol in recent historical context. This huge achievement in AI is based largely on CI methods, including DCNNs, SL from expert games, RL, the use of the

Google DeepMind introduced a new approach that combined MCTS with deep learning in their program AlphaGo, which subsequently broke this four-stone handicap barrier in the recent competition with Lee Sedol (9P) in Korea, in March 2016.

**Comment 2. Overall Comments from Human Go Players**

**Overall Comments by Ming-Wan Wang (9P/Japan)**
- Playing Go is not just a matter of achieving victory, but the knowledge involved and enjoyment of the game are also important. Currently, it seems to be difficult for top professional Go players to beat AlphaGo. However, the purpose of playing against human Go players is not simply for the victory itself, but also for communicating and sharing the game-playing process with each other. Go is an interesting game that leads to a lot of profound discoveries. Sometimes, we pay too much attention to improving our strength in playing skill, neglecting other things which we should focus on. We should explore Go from another perspective to enhance our enjoyment of the game.
- Each side (White or Black) needs to play at least over 100 moves during each Go game. Humans might lose the game from only one mistake in a decisive battle. In addition to the disadvantages in physical and mental conditions, humans do not adopt strategies from a whole-game perspective. A human Go player could manage to find a critical move to increase the winning rate, but fail to evaluate all of the possible moves. Nevertheless, an inevitable result is a human Go player’s subjective thoughts, which perhaps influence him/her to correctly analyze game situations.

**Overall Comments by Ping-Chiang Chou (6P/Taiwan)**
- AlphaGo’s strengths: AlphaGo shows very strong judgement both for the whole board and for local situations. Additionally, AlphaGo is also able to accurately calculate territory/points.
- AlphaGo’s weaknesses: When the situation favors AlphaGo, its playing style will become conservative. AlphaGo is capable of strong local computation but it does not seem to choose the most optimal move to play, for example, the fight in the bottom right corner on match No. 5 with Lee Sedol. It tends to avoid making kos if there are other lines of thinking to choose from.

**Overall Comments by Tai-Hsiung Yang (7D/Taiwan)**
- The aim of the computer Go program is to pursue the maximum probability to win instead of winning the maximum amount of territory. That is why AlphaGo sometimes loses some points by playing irrational moves when it apparently takes the lead. Human Go players suffer through terrible psychological stress when playing with such a strong computer Go program as AlphaGo. On the other hand, AlphaGo can also help find blind points that are easily missed by humans.
- It is not easy to differentiate between the middle game and the endgame. Sometimes, the endgame will start after the fighting. Other times, the fighting and the endgame will simultaneously happen. The computer Go program's computational speed is several times more powerful than the human Go player’s. Accordingly, developing a software for Go training is absolutely useful for strengthening professional Go players’ endgame skill.
- The game of Go contains a lot of the joy and philosophy of life; it is not just a game concerned with winning, but a cultural legacy that cultivates human talent.
value network and policy network, and MCTS. The strength of AlphaGo, especially as measured against the other computer Go programs, is absolutely amazing. Playing with contemporary computer Go programs like Crazy Stone, Zen, Pachi, Fuego, and GouGo, with no handicaps on either side, the single-machine version of AlphaGo was able to win 494 games out of 495 in total, while the distributed version of AlphaGo won all games against these competing programs [14].

As an alternative method of evaluation, the DeepMind team used an Elo rating scale, where each player gets a numerical strength estimation computed from past game results. AlphaGo, human European champion Fan Hui, and the above listed competing programs were rated. The distributed version of AlphaGo had an Elo rating of 3140, the non-distributed version was at 2890, and Fan Hui was at 2908 [14]. Since then, AlphaGo’s playing strength has grown even stronger. At the time of writing, according to an online ranking [22] curated by Go programmer and author of Crazy Stone, Rémi Coulom, distributed AlphaGo’s Elo rating is 3590 (the second highest rating in the world), where Ke Jie (3624) and Lee Sedol (3525) are ranked as the first and fourth highest Elo rating in the world, respectively. It is worth noting that the Elo rating system computed in [22] is not exactly the same as the one used in [14]. The ratings are given here only for the benefit of establishing some intuition for AlphaGo’s progress. In addition to the professional players’ comments on the five games between AlphaGo and Lee Sedol, this article also provides overall comments from Ming-Wang Wang (9P/Japan), Ping-Chiang Chou (6P/Taiwan), and Tai-Hsiung Yang (7D/Taiwan, director of Haifong Weiqi Academy, Taiwan) in Comment 2. AlphaGo’s victory over the world champion Lee Sedol in Mar. 2016 will be marked in history as a remarkable achievement. However, this would not have been possible without the considerable time and effort of countless contributors to computer Go in the past. AlphaGo’s impact will almost assuredly popularize and improve Go learning worldwide, especially if a personalized version with reduced hardware costs becomes available. Finally, AlphaGo’s performance was truly astonishing, and will undoubtedly be a continued source of inspiration for professional Go players and AI researchers around the world.

Acknowledgements
The authors would like to thank the Ministry of Science and Technology of Taiwan for its financial support under the grant MOST 105-2919-I-001-04A1, MOST 104-2221-E-005-015, and MOST 104-2622-E-005-001-CC2. Additionally, the authors also would like to thank 1) Mr. Ti-Rong Wu (National Chiao Tung University, Taiwan) for providing the resource estimate in generating self-play game positions (see Section II); 2) Mr. Sheng-Shu Chang (President of Click108 Company, Taiwan) for his financial support on past human vs. computer Go competitions @ IEEE CIS flagship conferences; and 3) Dr. Olivier Teytaud and INRIA TAO team members as well as Taiwanese team members and National Center for High-Performance Computing (NCHC) under the grant NSC 99-2923-E-024-003-MY3. Finally, we would like to thank the anonymous referees for their constructive and useful comments.

References
IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology
23-25 August 2017 – Manchester, UK

CIBCB 2017

The IEEE Conference on Computational in Bioinformatics and Computational Biology (IEEE CIBCB) has become a major technical event in the field of Computational Intelligence (CI) and its application to problems in bioinformatics, computational biology, and biomedical engineering. IEEE CIBCB 2017 provides a global forum for academic and industrial scientists from computer science, biology, chemistry, medicine, mathematics, statistics, and engineering, to discuss and present their latest research findings from theory to applications.

IEEE CIBCB will be held at the INNSIDE Hotel, Manchester, UK. Manchester is a city which has grown from the first industrial city and the home of the first programmable computer, to be the central hub of the “Powerhouse of the North” of England – an influential, modern, knowledge-based city amongst the top European locations for science, healthcare, education, business and leisure.

2017 marks also the return of IEEE CIBCB to Europe, having been hosted here only once before (Paris, 2011) since its foundation in 2004. We believe that the UK and Manchester is the right location to expand the popularity of this exciting research field and the CIBCB conference itself in this part of the world.

Call for Papers

Papers for IEEE CIBCB 2017 should be submitted electronically through the Conference website at www.cibcb2017.org, and will be refereed by experts in the fields and ranked based on the criteria of originality, significance, quality and clarity. Topics of interest include but are not limited to:

CI in Bioinformatics
- Evolution, phylogeny, comparative genomics
- Gene expression array analysis
- Metabolic pathway analysis
- MicroRNA analysis
- Molecular sequence alignment and analysis
- Pattern recognition/data mining/optimization methods in Bioinformatics
- Visualization of large biological datasets

CI in Computational Biology
- Systems and synthetic biology
- Structure prediction and folding
- Modelling, simulation, and optimization of biological systems
- Biological network reconstruction/robustness/evolvability
- Epigenomics

CI in Biomedical Engineering, Bioprocessing and Healthcare Informatics
- Medical imaging and pattern recognition
- Biomedical data modeling/data mining/model parameterization
- Parallel/high performance computing
- Big data analysis and tools for biological and medical data
- Biomarker discovery and development
- Health data acquisition/analysis/mining
- Healthcare information systems/knowledge representation/reasoning
- Personalized medicine and treatment optimization
- Biopharmaceutical manufacturing
- Closed-loop optimization methods/platforms

Call for Special Session Proposals

IEEE CIBCB 2017 solicits proposals for special sessions within the technical scope of the conference. Special session proposals should include the session title, a brief description of the scope, contact information and brief CV of the organizers. Inquiries regarding special session proposals should be addressed to the Special Session Chair.

Call for Tutorials

IEEE CIBCB 2017 will feature tutorials covering fundamental and advanced topics in computational intelligence and its applications to problems within the scope of the conference. A tutorial proposal should include title, outline, enrolment, and presenter/organizer biography. Inquiries regarding tutorial proposals should be addressed to the Tutorial Chair.

Call for Competitions

IEEE CIBCB 2017 enters all accepted posters into a Best Poster Competition. All conference attendees will have the chance to vote in the Competition on site. The best judged poster will win a prize and be featured on the IEEE CIBCB 2017 website. Inquiries regarding the Competition should be addressed to the Competition Chair.

For additional information please contact Richard Allmendinger at richard.allmendinger@manchester.ac.uk
Joint 8th International Conference on Soft Computing and Intelligent Systems and 17th International Symposium on Advanced Intelligent Systems (SCIS&ISIS 2016)
Place: Sapporo, Japan
General Chairs: Takahiro Yamanoi, Byung-Jae Choi
Website: http://scis2016.j-soft.org/

4th International Symposium on Computational and Business Intelligence (ISCBI 2016)
September 5–7, 2016
Place: Olten, Switzerland
General Chair: Suash Deb
Website: http://www.iscbi.com/

Sixth Joint IEEE International Conference on Developmental Learning and Epigenetic Robotics (IEEE ICDL-EpiRob 2016)
September 19–22, 2016
Place: Cergy-Pontoise, France
General Chairs: Minoru Asada, Philippe Gaussier
Website: http://www.icdl-epirob.org/

2016 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (IEEE CIBCB 2016)
October 5–7, 2016
Place: Chiang Mai, Thailand
General Chair: Sassanee Auephanwiriyakul
Website: http://cibcb2016.org/

Sixth Workshop on Brain-Machine Interface Systems (BMI Workshop 2016)
October 9–12, 2016
Place: Budapest, Hungary
General Chair: Michael H. Smith
Website: http://www.smc2016.org

2016 IEEE Conference on Data Science and Advanced Analytics (IEEE DSAA 2016)
October 17–19, 2016
Place: Montreal, Canada
General Chairs: Osmar R. Zaiane, Stan Matwin
Website: https://www.ualberta.ca/~dsaa16

11th International Workshop on Semantic and Social Media Adaptation and Personalization (SMAP 2016)
October 20–21, 2016
Place: Thessaloniki, Greece
General Chairs: Ioannis Anagnostopoulos, Iraklis Parasakis
Website: http://www.smap2016.org/

October 20–22, 2016
Place: Hermosillo, Sonora, Mexico
General Chair: Juan Manuel Ramos Arreguin
Website: http://www.mecamex.org/MAIS/2016

2016 IEEE Latin American Conference on Computational Intelligence (IEEE LA-CCI 2016)
November 2–4, 2016
Place: Cartagena de Indias, Colombia
General Chair: Alvaro David Orjuela Cañón
Website: http://la-cci.org

12th IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF 2016)
November 7–9, 2016
Place: Hanoi, Vietnam
General Chairs: Vincenzo Piuri, Tu Bao Ho
Website: http://rivf2016.tlu.edu.vn

Third International Conference on Behavioral, Economic and Socio-Cultural Computing (BESC 2016)
November 11–13, 2016
Place: Durham, USA
General Chair: Paul Wang
Website: http://www.besc-conf.org/2016

8th International Conference on Brain-Inspired Cognitive Systems (BICS 2016)
November 28–30, 2016
Place: Beijing, China
General Co-Chairs: Cheng-Lin Liu, Amir Hussain
Website: http://bii.ia.ac.cn/bics-2016/index.html
“Game competitions may involve different player roles and be score-based rather than win/loss based. This raises the issue of how best to draw opponents for matches in ongoing competitions, and how best to rank the players in each role. An example is the Ms Pac-Man versus Ghosts Competition which requires competitors to develop software controllers for either or both protagonists: participants may develop controllers to take charge of the game’s characters or to best score against the opponent in a series of pair-wise matches. The authors analyze how many games are needed under two popular ranking algorithms, Glicko and Bayes Elo, before one can infer the strength of the players, according to the proposed solution concept, without performing an exhaustive evaluation. The authors show that Glicko should be the method of choice for online score-based game competitions.”

**IEEE Transactions on Cognitive and Developmental Systems**


Digital Object Identifier: 10.1109/TAMD.2015.2441375

“In this paper, the authors propose an active learning approach applied to a music performance imitation scenario. The humanoid robot iCub listens to a human performance and then incrementally learns to use a virtual musical instrument in order to imitate the given sequence. This is achieved by first learning a model of the instrument, needed to locate where the required sounds are heard in a virtual keyboard laid out in a tactile interface. Then, a model of its body capabilities is also learnt, which serves to establish the likelihood of success of the actions needed to imitate the sequence of sounds and to correct the errors made by the underlying kinematic controller. It also uses self-evaluation stages to provide feedback to the human instructor, which can be used to guide its learning process.”
IEEE Media Advertising Sales Offices

James A. Vick
Sr. Director, Advertising Business
+1 212 419 7767; Fax: +1 212 419 7589
jv.ieemedia@ieee.org

Marion Delaney
Advertising Sales Director
+1 415 863 4717; Fax: +1 415 863 4717
md.ieemedia@ieee.org

Mark David
Sr. Manager Advertising & Business Development
+1 713 265 6473; Fax: +1 732 981 1855
m.david@ieee.org

Mindy Belfer
Advertising Sales Coordinator
+1 732 562 3937; Fax: +1 732 981 1855
m.belfer@ieee.org

Product Advertising
Mid-Atlantic
Lisa Rinaldo
+1 732 772 0160; Fax: +1 732 772 0164
lr.ieemedia@ieee.org
NY, Nj, PA, DE, MD, DC, KY, WV

New England/South Central/
Eastern Canada
Jody Estabrook
+1 774 283 4528; Fax: +1 774 283 4527
j.ieemedia@ieee.org
CT, ME, VT, NH, MA, RI, AR, LA, OK, TX.
CANADA: Nova Scotia, Prince Edward Island,
Newfoundland, New Brunswick, Quebec

Southwest
Cathy Flynn
+1 770 645 2944; Fax: +1 770 993 4423
cf.ieemedia@ieee.org
VA, NC, SC, GA, FL, AL, MS, TN

Midwest/Central Canada
Dave Jones
+1 708-442-5633; Fax: +1 708-442-7620
dj.ieemedia@ieee.org
IL, IA, KS, MN, MO, NE, ND, SD,
WI, OH. CANADA: Manitoba,
Saskatchewan, Alberta

Midwest/Ontario, Canada
Will Hamilton
+1 269 381 2156; Fax: +1 269 381 2556
wh.ieemedia@ieee.org
IN, MI, CANADA: Ontario

West Coast/Mountain States/
Western Canada
Marshall Rubin
+1 818 888 2407; Fax: +1 818 888 4907
mr.ieemedia@ieee.org
AZ, CO, HI, NM, NV, UT,
CA, AK, ID, MT, WY, OR, WA
CANADA: British Columbia

Europe/Africa/Middle East/Asia/
Far East/Pacific Rim
Michael O’Kane
+1 44 1875 825 700; Fax: +1 44 1875 825 701
mo.ieemedia@ieee.org
Europe, Africa, Middle East, Asia, Far East,
Pacific Rim, Australia, New Zealand

Recruitment Advertising
Mid-Atlantic
Lisa Rinaldo
+1 732 772 0160; Fax: +1 732 772 0164
lr.ieemedia@ieee.org
CT, NY, NJ, PA, DE, MD, DC, KY, WV

Midwest/Central Canada
Dave Jones
+1 708 442 5633; Fax: +1 708 442 7620
dj.ieemedia@ieee.org
IL, IA, KS, MN, MO, NE, ND, SD,
WI, OH. CANADA: Manitoba,
Saskatchewan, Alberta

Midwest/Ontario, Canada
Will Hamilton
+1 269 381 2156; Fax: +1 269 381 2556
wh.ieemedia@ieee.org
IN, MI, CANADA: Ontario

Europe/Africa/Middle East/Asia/
Far East/Pacific Rim
Michael O’Kane
+1 44 1875 825 700; Fax: +1 44 1875 825 701
mo.ieemedia@ieee.org
Europe, Africa, Middle East, Asia, Far East,
Pacific Rim, Australia, New Zealand

Recruitment Advertising
Mid-Atlantic
Lisa Rinaldo
+1 732 772 0160; Fax: +1 732 772 0164
lr.ieemedia@ieee.org
CT, NY, NJ, PA, DE, MD, DC, KY, WV

New England/Eastern Canada
Liza Reich
+1 212 419 7578; Fax: +1 212 419 7589
e.reich@ieee.org
ME, VT, NH, MA, RI. CANADA:
Nova Scotia, Prince Edward Island,
Newfoundland, New Brunswick, Quebec

Southeast
Cathy Flynn
+1 770 645 2944; Fax: +1 770 993 4423
cf.ieemedia@ieee.org
VA, NC, SC, GA, FL, AL, MS, TN

Midwest/South Central/Central Canada
Darcy Giovangi
+1 224 616 3034; Fax: +1 847 729 4269
dg.ieemedia@ieee.org
AR, IA, TX, OK, IL, IN, IA, KS, MI,
MN, NE, ND, SD, OH, WI, MO.
CANADA: Ontario, Manitoba,
Saskatchewan, Alberta

West Coast/Mountain States/
Southwest/Asia
Tim Matteson
+1 310 836 4064; Fax: +1 310 836 4067
tm.ieemedia@ieee.org
AK, AZ, CO, HI, ID, MT, NM,
NV, OR, UT, WA, WY.
CANADA: British Columbia

Europe/Africa/Middle East
Michael O’Kane
+1 44 1875 825 700
mo.ieemedia@ieee.org
Europe, Africa, Middle East

Digital Object Identifier 10.1109/MCI.2016.2565105
IEEE World Congress on Computational Intelligence
08 - 13 July 2018, Rio de Janeiro, Brazil

The IEEE World Congress on Computational Intelligence (IEEE WCCI) is the largest technical event in the field of computational intelligence. The IEEE WCCI 2018 will host three conferences: The 2018 International Joint Conference on Neural Networks (IJCNN 2018), the 2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2018), and the 2018 IEEE Congress on Evolutionary Computation (IEEE CEC 2018) under one roof. It encourages cross-fertilization of ideas among the three big areas and provides a forum for intellectuals from all over the world to discuss and present their research findings on computational intelligence.

IEEE WCCI 2018 will be held at the Windsor Barra Convention Centre, Rio de Janeiro, Brazil. Rio de Janeiro is a wonderful and cosmopolitan city, ideal for international meetings. Rio boasts fantastic weather, savory cuisine, hospitable people, and modern infrastructure.

IJCNN is the flagship conference of the International Neural Network Society and the IEEE Computational Intelligence Society. It covers a wide range of topics in the field of neural networks, from biological neural network modeling to artificial neural computation.

FUZZ-IEEE is the foremost conference in the field of fuzzy systems. It covers all topics in fuzzy systems, from theory to applications.

IEEE CEC is the leading event in the field of evolutionary computation, and covers all topics in evolutionary computation from theory to applications.

Call for Papers

Papers for IEEE WCCI 2018 should be submitted electronically through the Congress website at www.wcci2018.org, and will be refereed by experts in the fields and ranked based on the criteria of originality, significance, quality, and clarity.

Call for Tutorials

IEEE WCCI 2018 will feature pre-Congress tutorials, covering fundamental and advanced topics in computational intelligence. A tutorial proposal should include title, outline, expected enrollment, and presenter/organizer biography. Inquiries regarding tutorials should be addressed to Tutorials Chairs.

Call for Special Session Proposals

IEEE WCCI 2018 solicits proposals for special sessions within the technical scope of the three conferences. Special sessions, to be organized by internationally recognized experts, aim to bring together researchers in special focused topics. Cross-fertilization of the three technical disciplines and newly emerging research areas are strongly encouraged. Inquiries regarding special sessions and proposals should be addressed to Special Sessions Chairs.

Call for Competition Proposals

IEEE WCCI 2018 will host competitions to stimulate research in computational intelligence. A competition proposal should include descriptions of the problem(s) addressed, evaluation procedures, and a biography of the organizers. Inquiries regarding competitions should be addressed to the Competitions Chair.

Important Dates

- 1st February 2018: Paper Submission Deadline
- 15 December 2017: Tutorials & Special Sessions Proposal Deadline
- 1st April 2018: Paper Acceptance Notification Date
- 1st May 2018: Final Paper Submission
- 8-13 July 2018: IEEE WCCI 2018, Rio de Janeiro, Brazil

Digital Object Identifier 10.1109/MCI.2016.2585319
Data driven scientific discovery approach has already been agreed to be an important emerging paradigm for computing in areas including social, service, Internet of Things (or sensor networks), and cloud. Under this paradigm, Big Data is the core that drives new researches in many areas, from environmental to social. There are many new scientific challenges when facing this big data phenomenon, ranging from capture, creation, storage, search, sharing, analysis, and visualization. The complication here is not just the storage, VO, query, and performance, but also the integration across heterogeneous, interdependent complex data resources for real-time decision-making, collaboration, and ultimately value co-creation. Data sciences encompass the larger areas of data analytics, machine learning and managing big data. Data analytics has become essential to glean a deep understanding of large data sets and to convert data into actionable intelligence. With the rapid growth in the volumes of data available to enterprises, Government and on the web, automated techniques for analyzing the data have become essential. The 2017 International Conference on Data Science and Advanced Analytics (DSAA'2017), fully sponsored by IEEE and technically sponsored by ACM, aims to provide a premier forum that brings together researchers, industry practitioners, as well as potential users of data science, big data and advanced analytics, to promote collaborations and exchange of ideas and experiences. We discuss new opportunities, and investigate the best actionable analytics framework for wide range of applications. The conference solicits experimental and theoretical works on data science and advanced analytics along with their application to real life situations.

**Topics of Interest**

General areas of interest to DSAA'2017 include but are not limited to:

- **Foundations**
  - Mathematical, probabilistic and statistical models and theories
  - Machine learning theories, models and systems
  - Knowledge discovery theories, models and systems
  - Manifold and metric learning
  - Deep learning
  - Scalable analysis and learning
  - Non-IIDness learning
  - Heterogeneous data/information integration
  - Data pre-processing, sampling and reduction
  - Dimensionality reduction
  - Feature selection, transformation and construction
  - Large scale optimization
  - High performance computing for data analytics
  - Architecture, management and process for data science

- **Data analytics, machine learning and knowledge discovery**
  - Learning for streaming data
  - Learning for structured and relational data
  - Latent semantics and insight learning
  - Mining multi-source and mixed-source information
  - Mixed-type and structure data analytics
  - Cross-media data analytics
  - Big data visualization, modeling and analytics
  - Multimedia/stream/text/visual analytics
  - Relation, coupling, link and graph mining
  - Personalization analytics and learning
  - Web online/social/network mining and learning
  - Structure/group/community/network mining
  - Cloud computing and service data analysis

- **Storage, retrieval and search**
  - Data warehouses, cloud architectures
  - Large-scale databases
  - Information and knowledge retrieval, and semantic search
  - Web/social/databases query and search
  - Personalized search and recommendation
  - Human-machine interaction and interfaces
  - Crowd sourcing and collective intelligence

- **Privacy and security**
  - Security, trust and risk in big data
  - Data integrity, matching and sharing
  - Privacy and protection standards and policies
  - Privacy preserving big data access/analytics
  - Social impact

- **Applications**
  - Best practices and lessons learned from both success and failure
  - Data-intensive organizations, business and economy Privacy preserving big data access/analytics
  - Quality assessment and interestingness metrics
  - Complexity, efficiency and scalability
  - Big data representation and visualization
  - Business intelligence, data-lakes, big-data technologies
  - Large scale application case studies and domain-specific applications

**Key Dates**

- **Early Registration:** Aug. 25, 2017
- **Notification of acceptance:** July 25, 2017
- **Camera-Ready:** Aug. 15, 2017
- **Final Registration:** Aug. 31, 2017
- **Final Conference:** Oct. 19, 2017

**Call for Papers**

- **Website:** [http://www.dslab.it.aoyama.ac.jp/dsaa2017/](http://www.dslab.it.aoyama.ac.jp/dsaa2017/)

**Sponsors**

- IEEE
- ACM

**Editors**

- Hidetsugu Matoda, Osaka Univ., Japan
- Fosca Giannotti, ISTI, Italy
- Tomoyuki Higuchi, ISM, Japan
- Rajesh Parekh, Facebook, USA
- Yoji Kiyota, NEXT Co., Ltd, Japan
- Satoshi Kurihara, The Univ. of Electro-Comm., Japan
- Hang Li, Huawei Technologies, Hong Kong
- Diane J. Cook, Washington State University, USA
- Philip S. Yu, University of Illinois at Chicago, USA

**Conference Proceedings**

- All accepted papers will be published by IEEE and included in the IEEE Xplore Digital Library. The conference proceedings will be submitted for EI indexing through INSPEC by IEEE. Accepted Long presentation papers will be invited to the J. Data Science and Analytics, Springer.

**Digital Object Identifier** 10.1109/MCI.2016.2572599