5-2021

Prediction, Recommendation and Group Analytics Models in the domain of Mashup Services and Cyber-Argumentation Platform

Md Mahfuzer Rahman
University of Arkansas, Fayetteville

Prediction, Recommendation and Group Analytics Models in the domain of Mashup Services and Cyber-Argumentation Platform.

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy in Engineering with a concentration in Computer Science

by

Md Mahfuzer Rahman
Bangladesh University of Engineering and Technology
Bachelor of Science in Computer Science and Engineering, 2014

May 2021
University of Arkansas

This dissertation is approved for recommendation to the Graduate Council.

Xiaoqing "Frank" Liu, Ph.D. Committee Co-Chair	Brajendra Nath Panda, Ph.D. Committee Co-Chair
Douglas Adams, Ph.D. Committee Member	Susan E. Gauch, Ph.D. Committee Member
Justin Zhan, Ph.D. Committee Member	Qinghua Li, Ph.D. Committee Member

Abstract

Mashup application development is becoming a widespread software development practice due to its appeal for a shorter application development period. Application developers usually use web APIs from different sources to create a new streamlined service and provide various features to end-users. This kind of practice saves time, ensures reliability, accuracy, and security in the developed applications. Mashup application developers integrate these available APIs into their applications. Still, they have to go through thousands of available web APIs and chose only a few appropriate ones for their application. Recommending relevant web APIs might help application developers in this situation. However, very low API invocation from mashup applications creates a sparse mashup-web API dataset for the recommendation models to learn about the mashups and their web API invocation pattern. One research aims to analyze these mashup-specific critical issues, look for supplemental information in the mashup domain, and develop web API recommendation models for mashup applications. The developed recommendation model generates useful and accurate web APIs to reduce the impact of low API invocations in mashup application development.

Cyber-Argumentation platform also faces a similarly challenging issue. In large-scale cyber argumentation platforms, participants express their opinions, engage with one another, and respond to feedback and criticism from others in discussing important issues online. Argumentation analysis tools capture the collective intelligence of the participants and reveal hidden insights from the underlying discussions. However, such analysis requires that the issues have been thoroughly discussed and participant's opinions are clearly expressed and understood. Participants typically focus only on a few ideas and leave others unacknowledged and under-discussed. This generates a limited dataset to work with, resulting in an incomplete analysis of

issues in the discussion. One solution to this problem would be to develop an opinion prediction model for cyber-argumentation. This model would predict participant's opinions on different ideas that they have not explicitly engaged.

In cyber-argumentation, individuals interact with each other without any group coordination. However, the implicit group interaction can impact the participating user's opinion, attitude, and discussion outcome. One of the objectives of this research work is to analyze different group analytics in the cyber-argumentation environment. The objective is to design an experiment to inspect whether the critical concepts of the Social Identity Model of Deindividuation Effects (SIDE) are valid in our argumentation platform. This experiment can help us understand whether anonymity and group sense impact user's behavior in our platform. Another section is about developing group interaction models to help us understand different aspects of group interactions in the cyber-argumentation platform.

These research works can help develop web API recommendation models tailored for mashup-specific domains and opinion prediction models for the cyber-argumentation specific area. Primarily these models utilize domain-specific knowledge and integrate them with traditional prediction and recommendation approaches. Our work on group analytic can be seen as the initial steps to understand these group interactions.

Acknowledgments

I want to thank my advisor Dr. Xiaoqing "Frank" Liu, for his continuous support and guidance throughout my Ph.D. program. He guided me academically, provided valuable insights, set standards, and supervised me in my research, contributing to producing and delivering research works presented in this dissertation.

Also, I want to thank Dr. Douglas Adams for advising me in my research. I learned many valuable social-psychological-behavioral pieces of knowledge from him, which shaped different research works presented in this dissertation.

I would like to thank Dr. Brajendra Nath Panda for his kind support in my Ph.D. program. He acted as my co-advisor and dissertation committee co-chair and advised me academically in my Ph.D. program's last two semesters.

I want to thank my fellow graduate students Joseph Sirrianni, and Najla Althuniyan, with whom I worked for an extended period on the Intelligent Cyber Argumentation Platform. I would also like to express my gratitude to my fellow lab mates Md Rakib Shahriar and S M Nahian Al Sunny; I worked with them on few research projects.

Lastly, I want to thank my committee members, Dr. Susan Gauch, Dr. Justin Zhan, and Dr. Qinghua Li, for their valuable inputs on my research and dissertation. Thank you all for your kind support and help. I appreciate it a lot.

Table of Contents

List of Figures

List of Published Papers

Chapter 2, 3, 4, and 5 are partial reproductions of published papers or accepted for publications on the following venues:

Chapter 2:

[1] M. M. Rahman, X. Liu, and B. Cao, "Web API Recommendation for Mashup Development Using Matrix Factorization on Integrated Content and Network-Based Service Clustering," in 2017 IEEE International Conference on Services Computing (SCC), pp. 225–232, doi: 10.1109/SCC.2017.36.

Chapter 3:

[2] M. M. Rahman, and X. F. Liu, "Integrated Topic Modeling and User Interaction Enhanced WebAPI Recommendation using Regularized Matrix Factorization for Mashup Application Development," in 2020 IEEE International Conference on Services Computing (SCC), Beijing, 2020, pp. 124-131.

Chapter 4 and 5:

[3] M. M. Rahman, X. F. Liu, J. W. Sirrianni, and D. Adams, “Cross-Issue Correlation based Opinion Prediction in Cyber Argumentation” in Argument & Computation Journal, 2021 (Accepted – In Press).[1]

[4] M. M. Rahman, J. W. Sirrianni, X. F. Liu, and D. Adams, “Predicting opinions across multiple issues in large scale cyber argumentation using collaborative filtering and viewpoint correlation,” in ninth International Conference on Social Media Technologies, Communication, and Informatics (SOTICS 2019), pp. 45-51.

[1] [3] is the expanded version of [4]

Chapter 1: Introduction

Mashup application development is a rapidly growing software development practice where application developers usually use web APIs from different sources to create a new streamlined service providing various features to end-users. Mashup development usually requires little programming knowledge. Data-centric and graphical user interface-based application development is one of the main reasons for the rapid increase of mashup-based software development. This lucrative process drew many well-known computer organizations in developing different mashup application editing tools. Mashup development encourages end-user software development. End users are often domain experts and with little programming knowledge. They develop applications collaboratively from existing sources rather than going through the lengthy software development cycle. Mashup applications also help with situational software development where a software application needs to be developed for only a small number of users satisfying their particular needs or for a specific task for a group or business corporation [1]. Due to the limited scope of these software applications, the only way these software applications would be cost-effective if they can be developed in a short period yielding a low development cost.

Due to the increasing adoption of service-oriented architecture by the web and business organizations, they publish their services as Application Program Interface (API). This enables mashup application developers to easily integrate these available APIs into their applications instead of writing these services independently. Currently, there are thousands of published web APIs available, which mashup application developers can consider for their application. ProgrammableWeb[1] is an API track-keeping repository. According to this site, as of May 2020,

1. https://www.programmableweb.com/

there are 22,992 APIs in different categories. Also, new APIs are rapidly added to their site, 2019 new APIs are being added at the ProgrammableWeb [2]. This situation is especially challenging for mashup application developers as they have to browse through thousands of available APIs and choose only a few for their application, which is not a feasible task for application developers. If we can recommend accurate and useful APIs to the application developers, it would be easier for them to find the appropriate APIs for their application.

However, these recommendation models face one common challenging issue: the sparsity of the entire dataset. On average, a mashup application only invokes three to five web-APIs, but there are thousands of web-APIs available for recommendation [3]. This low invocation from mashups generates a very sparse dataset for recommendation models to learn about individual mashup applications and identify meaningful information from the underlying data. As a result, recommendations are not often reasonably accurate. These recommendation models have only a few web APIs to learn about individual mashup application but a thousand web APIs to consider for the recommendation.

In our first research task, we developed a web API recommendation model for the mashup application. This approach uses a two-level topic modeling both from mashup's own content and content from its' network to identify similar mashup services together. Later, we utilized similar mashup services information via a matrix-factorization model to generate accurate and useful recommendations for mashup applications. Then, we analyzed two critical issues in mashup application development. First, mashup usually invokes very few APIs, which generates a sparse dataset and affects the web API recommendation models. Second, many mashups share various web APIs, and many web APIs are being used by a mashup in this domain, which can work as supplemental information in the recommendation process. We specially designed our second web

API recommendation model so that it can use the additional data and integrate them with the traditional web API invocation analysis to reduce the impact of low API invocations in web API recommendation. Also, this model uses two techniques sequentially to identify similar and related mashup and web APIs to reduce the sparsity of the initial dataset. Both of these models were evaluated using real mashup and web API dataset collected from ProgrammbleWeb.

In large-scale cyber argumentation platforms, participants express their opinions, engage with one another, and respond to feedback and criticism from others in discussing important issues online. Cyber argumentation platforms implement argumentation models to enforce an explicit discussion structure, such as Dung abstract frameworks [4], Issue-Based Information Systems (IBIS) [5], and Toulmin's model of argumentation [6]. These structures allow argumentation analysis tools to analyze the discussions effectively. Argumentation analysis tools can capture the participants' collective intelligence and reveal hidden insights from the underlying discussions. In this research domain, these tools have demonstrated the ability to evaluate and reveal hidden phenomena, such as identifying group-think [7], polarization [8], assessing argument validity [4], etc.

However, such analysis requires that the issues have been thoroughly discussed and participant's opinions are clearly expressed and understood. Participants typically focus only on a few ideas and leave others unacknowledged and under-discussed. This generates a limited dataset to work with, resulting in an incomplete analysis of issues in the discussion. This also hampers the individual and collective intelligence retrieval process and opinion analysis from the underlying discussion. Particularly a limited dataset with missing values affects the clustering or user grouping algorithms, and the resulting user groups introduce error and bias in different social phenomena analysis [9].

In our third research task, we developed a model for predicting participant's opinions on different ideas that they have not explicitly engaged. We use our argumentation platform, the Intelligent Cyber Argumentation System (ICAS), to collect user opinion on issues and predict the missing opinions. In our system, discussions take on a tree structure. Issues are the root of the conversation. Under an issue, there is a finite set of different positions that address the issue. We use a collaborative filtering model based on viewpoint correlation between positions and user opinion similarity to predict the user's missing opinion on a position.

Later, we focused on group interactions in the cyber-argumentation platform. Although participating users discuss different social and political issues in the platform, groups can be implicit within the discussion, which can impact the participating users and the collective discussion outcome. The social identity theory (SIT) [10] and the social identity model of deindividuation effects (SIDE) [11] are two of the social science theories. These two theories analyze different aspects of human behavior, such as how they perceive themselves, adjust their opinion, attitude, and behavior in an anonymous group setting, and how they behave towards the people within their in-group [10] and out-groups [10]. These theories are very useful in designing user behavior models and group interaction techniques in different online platforms. However, before using these theories in user opinion modeling and other argumentation phenomena analysis models, we first need to examine whether these theories are valid in an online discussion setting. In our fourth research task, we designed an experiment to analyze whether the anonymity in our platform and psychological group sense from a similar opinion influences users' behavior related to in-group and out-group activity as per the SIDE model in our platform. Also, we worked on two critical areas in cyber-argumentation. First, we developed different group interaction models for

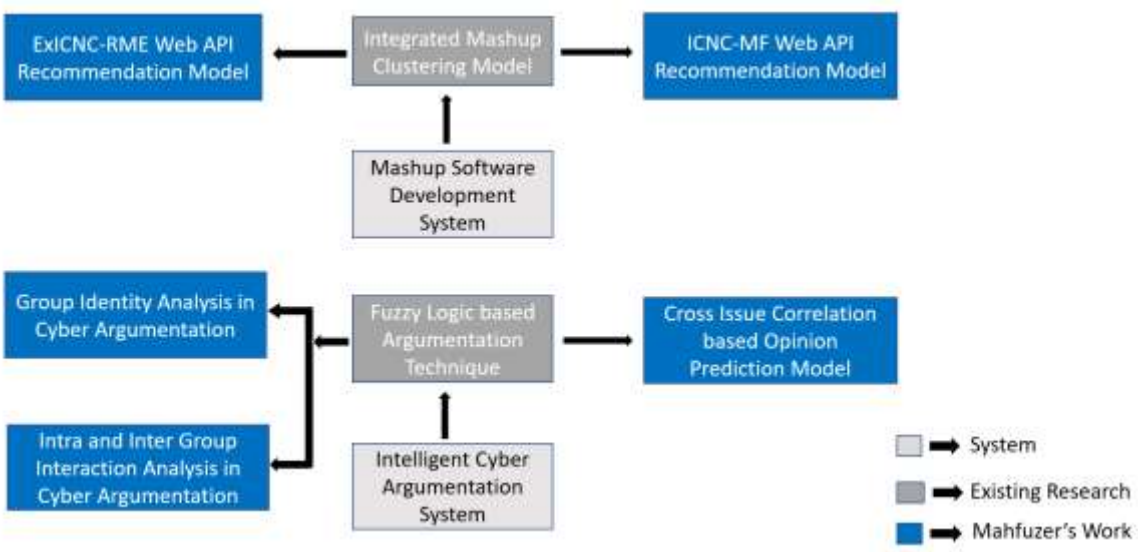

Figure 1. 1 Dissertation Defense Framework

the cyber-argumentation platform. Currently, there is no such model that can help us understand the impact of these group interactions at the individual and collective levels. We analyzed and developed models to understand how supportive or critical the group members are to each other and another model for understanding how supportive or critical the groups are to each other as a collective entity in the discussion.

Figure 1.1 gives an overall framework of this dissertation proposal. In the figure, we can see that we used the mashup clustering model in three of the developed web API recommendation models in this research task. This integrated closeting model was developed by [3], which we used in our two developed recommendation models. Also, fuzzy logic and argumentation techniques were developed in the prior research work by [12, 13, 14, 15]. These techniques were used to develop the Intelligent Cyber Argumentation System as a discussion platform. This platform's different datasets were used in the opinion prediction model and group analytics model.

1.1 Reference

[1] K.-B. Yue, "Experience on Mashup Development with End User Programming Environment," Journal of Information Systems Education, vol. 21, no. 1, Nov. 2019, [Online]. Available: https://aisel.aisnet.org/jise/vol21/iss1/10.

[2] "APIs show Faster Growth Rate in 2019 than Previous Years | ProgrammableWeb." [Online]. Available: https://www.programmableweb.com/news/apis-show-faster-growth-rate-2019-previous-years/research/2019/07/17. [Accessed: 04-Mar-2020].

[3] B. Cao, X. F. Liu, M. M. Rahman, B. Li, J. Liu, and M. Tang, "Integrated Content and Network-Based Service Clustering and Web APIs Recommendation for Mashup Development," IEEE Transactions on Services Computing, vol. 13, no. 1, pp. 99–113, Jan. 2020, doi: 10.1109/TSC.2017.2686390.

[4] P. M. Dung, "On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games," Artificial Intelligence, vol. 77, no. 2, pp. 321–357, Sep. 1995.

[5] W. Kunz and H. W. J. Rittel, "Issues as elements of information systems," Inst. Urban and Regional Devt., Univ. Calif. at Berkeley, 1970.

[6] S. E. Toulmin. 1958. The Uses of Argument. Cambridge, UK: University Press, 1958.

[7] M. Klein, "The CATALYST Deliberation Analytics Server," Social Science Research Network, Rochester, NY, SSRN Scholarly Paper, Nov. 2015.

[8] J. Sirrianni, X. Liu, and D. Adams, "Quantitative Modeling of Polarization in Online Intelligent Argumentation and Deliberation for Capturing Collective Intelligence," 2018 IEEE International Conference on Cognitive Computing (ICCC), pp. 57–64, 2018.

[9] S. Zhang, J. Zhang, X. Zhu, Y. Qin, and C. Zhang, "Missing Value Imputation Based on Data Clustering," in Transactions on Computational Science I, M. L. Gavrilova and C. J. K. Tan, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 128–138.

[10] Tajfel, H., Turner, J.C.: The Social Identity Theory of Intergroup Behavior. In: Jost, J.T. and Sidanius, J. (eds.) Political Psychology. pp. 276–293. Psychology Press (2004)

[11] Reicher, S.D., Spears, R., Postmes, T.: A Social Identity Model of Deindividuation Phenomena. European Review of Social Psychology. 6, 161–198 (1995). doi:10.1080/14792779443000049

[12] X. Liu, E. Khudkhudia, L. Wen and V. Sajja, An Intelligent Computational Argumentation System for Supporting Collaborative Software Development Decision Making, 2010, pp. 167–180. doi:10.4018/978-1-60566-758-4.ch009.

[13] S. Sigman and X.F. Liu, A computational argumentation methodology for capturing and analyzing design rationale arising from multiple perspectives, Information & Software Technology 45 (2003), 113–122. doi:10.1016/S0950-5849(02)00187-8.

[14] X.F. Liu, S. Raorane and M.C. Leu, A Web-based Intelligent Collaborative System for Engineering Design, in: Collaborative Product Design and Manufacturing Methodologies and Applications, W.D. Li, C. McMahon, S.K. Ong and A.Y.C. Nee, eds, Springer Series in Advanced Manufacturing, Springer London, London, 2007, pp. 37–58. ISBN 978-1-84628-802-9. doi:10.1007/978-1-84628-802-9_2.

[15] X.F. Liu, E.C. Barnes and J.E. Savolainen, Conflict Detection and Resolution for Product Line Design in a Collaborative Decision Making Environment, in: Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work, CSCW '12, ACM, New York, NY, USA, 2012, pp. 1327–1336, event-place: Seattle, Washington, USA. ISBN 978-1-4503-1086-4. doi:10.1145/2145204.2145402.

Chapter 2: A Web API Recommendation Model for Mashup development using Matrix Factorization with Integrated Content and Network-Based Service Clustering Technique

2.1 Abstract

Finding appropriate web APIs to develop mashup services is becoming difficult because of the increasing number of web APIs offered from different sources. If we can recommend relevant web APIs for a mashup service based on its requirements, it will help software developers to find suitable APIs easily instead of searching from thousands of web APIs. Although there are many existing methods to recommend web APIs for mashup services, their recommendation accuracies and diversities are still not high. We will present a novel approach in this paper to produce better web API recommendation results in terms of accuracy and diversity. It is a matrix factorization based API recommendation method for Mashup services. It uses a two-level topic model for clustering Mashup services. We used a dataset from programmableWeb to perform experiments and compared the results of our method with other existing methods. Its evaluation results show that our matrix factorization based recommendation archives better API recommendation accuracy and diversity for Mashup services.

2.2 Introduction

Mashup technology has become very popular in recent years, which allows software developers to compose web APIs from multiple sources to create a new single service or application. There are several online repositories of mashup services and web APIs available, such as ProgrammableWeb, myExperiment, and Biocatalouge. Users can choose existing web APIs or mashup services to create their own mashup services according to their needs.

However, the selection of the appropriate web APIs and mashup services from these repositories is a challenging task since the number of available web APIs and mashup services is huge. For example, ProgrammableWeb published 15,788 web APIs and 7828 mashups under more than 400 categories. If a developer wants to build a mashup related to messaging, programmableWeb search result returns 1217 web APIs and 472 mashups. It is not an easy task to go through these lists of search results and select the desired APIs and mashup services.

Many researchers worked on web service recommendations. Several researchers considered the similarity between the user requirements and capabilities of the available services in their recommendation methods [2, 6-8]. Other researchers used QoS (Quality of Service) based service recommendation via estimating QoS values for similar users or items and recommended services to users accordingly [2-3, 9-15]. Many researchers used relationships among services for service recommendation [4, 16-19]. In addition, many other researchers used a combination of two or many of the above methods to recommend web services [5, 20-23].

There are many existing methods to recommend web APIs for mashup development. C. Li et. al. [22] recommended APIs using a relational topic model and popularity of web APIs for new mashup development. S. R. Chowdhury et. al. [37] used the information of API input, output, and mashup structure to discover the composition pattern of mashups and recommended composition knowledge. H. Elmeleegy et al. [36] presented a mashup advisor who takes a partially complete mashup and shows possible outputs. The user selects the desired output, and the mashup advisor recommends the best services to achieve that output. R. Torres et al. [34] presented an API recommendation technique that integrated popular APIs with a search mechanism to recommend relevant APIs.

An existing investigation [24] shows that most mashup services contain maximal three web APIs. If a user wants to discover more APIs from existing mashup composition, users would not be able to find more than three Web APIs from a single mashup service. It is desirable to identify clusters of similar Mashup services and expand the service space for API recommendation in the mashup service development. Several researchers noted this issue and used service clustering in recommendation [1, 24]. However, the mashup services are related to each other via invoking common APIs, descriptive tags, etc. These relationships were not taken into consideration by them.

The accuracy and diversity of the existing API recommendation methods for the development of mashup services are still not satisfactory, even though progress was made. Most existing API recommendation methods for mashup development do not consider the diversity of recommendation results. They only focus on using popular APIs. If a new mashup is created using only popular web APIs based on historical usage, much less used but useful API may not be recommended [24]. A recommendation result should include both popular and less used web APIs to expand the service discovery space to find APIs to meet up user's requirements. Xia et al. [1] used a method of ranking Web APIs in each service category and diversify the recommendation results.

We propose matrix factorization based Web API Recommendation for Mashup development using Integrated Content and Network-Based Service Clustering (ICNC) [33]. Our model incorporates mashup service relationships to improve accuracy and diversity of API recommendation results for mashup development. Our experimental results show that our matrix factorization based recommendation achieves better accuracy and diversity than other existing methods.

2.3 Method Overview

2.3.1 Framework

we developed a framework of web API recommendation based on mashup service clustering for new mashup development. B. Cao et al. [33] presented ICNC based mashup clustering, which we used to cluster mashup services in this paper. Once we have clustering results, we applied a matrix factorization method to recommend web APIs for each mashup category. The Overview of our framework is shown in fig. 2.1.

ICNC method first collects service data and extracts the content feature vector, which represents mashup services. Then it builds a network of services based on relationships among them and uses a two-level topic model to identify functional topics of mashup services. It ranks them and selects those mashup services with the above similarity in a cluster. Then it applies the Agnes algorithm for hierarchical clustering and merging some similar clusters together.

We associated each mashup cluster with the web APIs from its consisting mashup services and applied a matrix factorization algorithm to predict the recommendation values of missing APIs. Then we integrated the recommended result with popular APIs and ranked them. Finally, we recommended top R web APIs for each mashup category. This list can be used as a recommendation for building a new mashup service.

2.3.2 Mashup Service Clustering

At first, the ICNC method creates a mashup service content document by collecting functional information of mashup services, including its name, category, typical description, web APIs, and tags. This document works as a complete description of mashup services than the typical one and is used to extract core feature vectors of mashup services. At first, a natural language processing

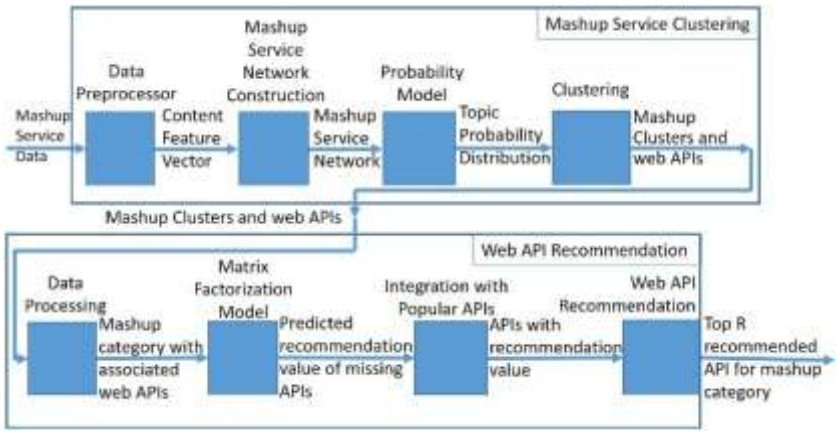

Figure 2. 1 Mashup Service Clustering and Web API Recommendation Framework

toolkit NLTK is used to divide sentences of mashup service description into words, and an initial feature vector is created. Then symbols and words like a, of, +, - etc. are removed as they do not contribute to characterize and compare feature words. Usually, nouns, adjectives, or verbs are meaningful feature words. Then Porter Stemmer in the NLTK toolkit is used to extract the stemming of all words to produce a new feature vector.

Mashup services are implicitly related to each other. This relationship can be assessed based on how many common APIs they invoke and how many same tags are used to mark them. ICNC method uses this information to build a mashup service network (MSN), which represents their correlation. Jaccard similarity coefficient is used to measure the edge weight or the similarity value between two mashup services using the following equation.

$$W(MS_i, MS_j) = \lambda_1 * \frac{|API(MS_i) \cap API(MS_j|}{|API(MS_i) \cup API(MS_j|} + \lambda_2 * \frac{|TAG(MS_i) \cap TAG(MS_j|}{|TAG(MS_i) \cup TAG(MS_j|} \qquad (1)$$

Here MS_i and MS_j are two mashup service nodes in MSN which is an undirected network graph to represent mashup services. W (MS_i, MS_j) represents the edge weight or the similarity between MS_i and MS_j. API(MS_i) and API(MS_j) represents APIs invoked by MS_i and MS_j. TAG(MS_i) and TAG(MS_j) represents TAGs used to mark MS_i and MS_j. λ_1 and λ_2 are preference over APIs and Tags where $\lambda_1 + \lambda_2 = 1$.

Mashup service topics are distributed at two different levels: Content and network level. One sub- model processes all mashup service documents at the content level. Then a mashup service network is built from these documents at the network level. Then topic distribution for each mashup service from all the linked mashup services is incorporated into its topic distribution at the content level. Therefore, there will be topics for each mashup services from two parts, one from its own and another from linked mashup services.

At the network level, the following operations are performed in the ICNC method to calculate the topic distribution of all linked mashup services for a particular mashup service.

- *For each linked(directly/indirectly) mashup service MS_j in L_{MS} where L_{MS} represents all linked mashup service of MS.*
  - *For i^{th} word in MS_j*
  1. *Select a topic Z_{ji} from MS_j's topic distribution using p ($z|MS_j, \theta_{MSj}$), θ_{MSj} is a distribution parameter which is calculated from Dirichlet distribution Dir(α).*
  2. *Select a word w_{ji} which follows multinomial distribution p ($w|z_{ji}, \varphi$) based on condition Z_{ji}.*

At the content level, the following operations are performed in the ICNC method to calculate the topic distribution for all mashup services. Here MS is a set of mashup services containing MS_1, MS_2, MS_3… MS_n.

- *For each mashup service document MS_s*
  - *For i^{th} word in MS_s*
  1. *Select a linked Mashup service document $L_{MS_{si}}$ from the multinomial distribution p($L_{MS}|MS_s$, Ψ), which is based on condition MS_s.*
  2. *Select a topic t_{si} from the topic distribution p($t|L_{MS_{si}}, \eta$) based on condition $L_{MS_{si}}$.*
  3. *Select a word $w_{si,}$ which follows multinomial distribution p ($w|t_{si}, \varphi$) based on condition t_{si}.*

Here Ψ is a selection co-efficient matrix that represents the probability of a mashup service at the network level that will be incorporated into another mashup service's content level. η is a topic selection coefficient matrix.

A link-level random walk on the mashup service network for mashup service document MS_s is performed in the ICNC method to calculate the matrix Ψ. For all linked (direct/indirect) mashup services MS_j of MS_s, a link probability score is associated, which is defined below:

$$P\left(L_{MS_{si}} = MS_j \middle| MS_s\right) = (1 - \beta) * (1 - \beta Q)^{-1} M \qquad (2)$$

Here, β is the probability that a random walk will not continue or stop at MS_j. M is an initial probability distribution vector where $m_j = 1/L(MS_j)$. Q is an adjacency matrix where q_{ij} is a random walk transition probability from MS_i to MS_j.

A topic level random walk is performed on the mashup service network in the ICNC method to calculate matrix η. A topic probability score vector P (MS_j, z) specified on topic z for each MS_j in L_{MS}. For all topics, a random walk is performed along with all linked (direct/indirect) mashup services in the mashup service network. In the network for a link from MS_i to MS_j, two types of transition probabilities are associated [defined in equation 3 and 4].

$$P(MS_j | MS_i, z_x) = \frac{1}{L(MS_i)} \qquad (3)$$

$$P\left(MS_j, z_x \middle| MS_i, z_y\right) = P(z_x \mid MS_j) * P\left(z_y \middle| MS_i\right) \qquad (4)$$

- P (MS_j | MS_i, z_x) is the topic-intra transition probability from MS_i to MS_j on topic z_x which is common between them;
- P (MS_j, z_x | MS_i, z_y) is the topic-inter transition probability from MS_i to MS_i on topics z_x and z_y which is different between them;
- $L(MS_i)$ represents the number of nodes directly connect to MS_i (degree of MS_i);

- P (z_x | MS_j) is the topic z_x generation probability by MS_j;
- P (z_y | MS_i) is the topic z_y generation probability by MS_i.

A parameter γ is used to control the preference on topic-intra and topic-inter transition probability during random walk. So we can find the topic probability score for a mashup service MS_j on topic z_x trough a topic level random walk.

$$P(MS_j, z_x) = \beta \sum_{MS_i:MS_i \to MS_j} \left[\gamma P(MS_j | MS_i, z_x) + (1 - \gamma) \frac{1}{|T|} \sum_{y \neq x} P(MS_j, z_x | MS_i, z_y)\right] + (1 - \beta) \frac{1}{|D|} P(z_x | MS_j) \qquad (5)$$

- P (MS_j, z_x) is topic probability score on z_x of MSj;
- P (z_x | MS_j), P (MS_j | MS_i, z_x), and P (MS_j, z_x | MS_i, z_y) are same as equation 3 and 4;
- |D| is the number of Mashup service documents in the *MSN*;
- |T| is the number of topic generated by MS_j.

Then the similarity among mashup services is computed using Kullback-Leibler (KL) and JS divergence algorithm [30]. Then the similarity result is integrated with K-Means and Agnes algorithms to cluster similar mashup services [30]. Topic probability distribution can be used to calculate the similarities between mashup service documents as topics can be mapped into document vector space, and topics represent the document materials. Following equation is used to measure the KL divergence:

$$D_{KL}(MS_i, MS_j) = \sum_{t=1}^{T} p_t \, ln \frac{p_t}{q_t} \qquad (6)$$

- D_{KL} (MS_i, MS_j) is the KL divergence value between MS_i and MS_j mashup services;
- t is a variable for common topic between MS_i and MS_j;
- T is the number of common topics between MS_i and MS_j;
- p_t is the probability of finding topic t in MS_i;

- q_t is the probability of finding topic t in MS_j.

As KL divergence is asymmetric, JS divergence is used to improve the similarity calculation between MS_i and MS_j based on the result from KL divergence.

$$D_{JS}(MS_i, MS_j) = \frac{1}{2}[D_{KL}\left(MS_i, \frac{MS_i+MS_j}{2}\right) + D_{KL}\left(MS_j, \frac{MS_i+MS_j}{2}\right)] \quad (7)$$

Here D_{JS} (MS_i, MS_j) is the similarity between MS_i, MS_j mashup service. Then K-means and Agnes algorithm are used to cluster the mashup services using their similarities value. First Mashup services are ranked and similar mashup services with above average value are selected using K-means algorithm. Then Agnes algorithm is used to hierarchically cluster these mashup services combining those mashup services with above threshold value.

2.3.3 Web API Recommendation based on Matrix Factorization

Since the number of APIs in each cluster is huge, we recommend top web APIs for each category. We used a matrix factorization method to recommend web APIs and showed that matrix factorization based recommendation works better than other existing baseline methods in terms of accuracy and diversity.

1) Matrix Factorization

Assume that we have a set of categories A and a set of web APIs B, and another matrix C of size |A| X |B|. Each value of C represents the popularity of an API in a category. The value in matrix C is normalized and $0 \leq c_{ij} \leq 1$. If there is K latent features, we need to find two different matrix P and Q such that P is a |A| X |K| matrix, Q is a |K| X |B| matrix, and $C \approx P \times Q^T$

To predict a missing value corresponding to each category a_i and web API b_j, we need to use the dot product of two vectors based on the following equation:

$$r'_{ij} = p_i^T q_j = \sum_{k=1}^{k} p_{ik} q_{kj} \quad (8)$$

We first initialize P and Q with random values and update the values in each iteration. In each iteration, we try to minimize the error. The error is calculated as how different the resultant dot product is with the original matrix. Two parameter α and β are used. Parameter α controls the rate of reaching to a minimum. Parameter β controls the size of P and Q for a better approximation of R. Using the following equation; we calculate the error:

$$e_{ij}^2 = (r_{ij} - \sum_{k=1}^{k} p_{ik} q_{kj})^2 + \frac{\beta}{2} \sum_{k=1}^{k} (||P||^2 + ||Q||^2) \qquad (9)$$

The square value of error is used because predicted value can be bigger or smaller than the original value. Using the gradient value of error, we can update the value of p_{ik} and q_{kj} using the following equations:

$$p'_{ik} = p_{ik} + \alpha (2e_{ij} q_{kj} - \beta p_{ik}) \qquad (10)$$

$$q'_{kj} = q_{kj} + \alpha (2e_{ij} p_{ik} - \beta q_{kj}) \qquad (11)$$

This procedure is conducted iteratively until the minimum error is reached.

2) Web APIs Recommendation for Mashup Clusters

After the values of missing APIs are calculated by matrix factorization, we can now recommend top web APIs for a mashup category. The value obtained from this algorithm is considered the predicted recommendation value for a web API. Then this result is integrated with popular choices of web APIs. After ranking the web APIs, the top R web APIs are recommended for each mashup category. The process of recommending web API is described in the below algorithm.

Algorithm: Recommendation of Web APIs

Input: *M= {M_1 M_2,…,M_K}, WA= {WA_1 WA_2,…, WA_N}*

// M represents a set of Mashup service clusters from Section 2.B; WA represents a set of Web APIs //

Output: *Top-T Web APIs Recommended for each Mashup category*

1. *C =|A| X |B| represents mashup services*
2. *and frequency of composing web APIs*
3. ***For*** *K=1 to M*
4. *P =|A| X |K|; Q = |K| X |B|*
5. *Initialize P and Q with random values*
6. ***Do {Update*** *each member of P and Q using formula*
7. *8,9,10,11;*
8. ***Calculate minimum error value*** e_{ij}
9. ***}*** *while (error value < Threshold value)*
10. $R \approx P \; X \; Q^T$ *// Q^T is the transpose matrix of Q//*
11. ***Compare R*** *with previously calculated R and*
12. *keep the best*
13. ***End For***
14. ***For*** *j = 1 to K // Number of mashup services is K*
15. ***For*** *l=1 to U // the number of Web APIs in Cluster*
16. M_j *is U //*
17. *Combine* WA_l*'s Prediction value with its Popularity value*
18. ***End For***
19. ***Rank*** M_j***'s*** *all composing web APIs;*
20. ***Add*** *Top-T Web APIs recommendations for* M_j *in Result*

21. End For

22. ***Return*** *Result // Result contains Top-T Web APIs Recommended for each Mashup cluster.*

In summary, the following process is used to recommend web APIs:

Using ICNC method similar mashup services are identified. Recommendation values for missing web APIs are predicted using matrix factorization method. This result is combined with popular web APIs in a mashup category. Top R web APIs are identified and recommended for each mashup category.

2.4 Experiments

2.4.1 Web API Recommendation

To evaluate the accuracy and diversity of the recommended APIs for each mashup category, we performed several experiments. They are based on the measurement of accuracy and diversity of the recommended APIs in terms of DCG (Discounted Cumulative Gain) and Hamming Distance (HMD) value. The following section describes these experiments and comparisons between our method and other existing baseline methods.

2.4.2 Experimental Dataset

From ProgrammableWeb site, we have collected 6960 real Mashup services with their related data and obtained Mashup service's name, category, description, tags, and web APIs. We have observed that variation in terms of the number of mashup services in categories is very large. As an example, category Mapping has 1038 Mashup services, while category Address has only 1 Mashup service. Since categories with a small number of mashup services contribute to poor clustering, we chose the top 20 categories containing 3929 Mashup services and 62078 words as the experimental

dataset. We developed a Mashup Service Network platform based on the experimental dataset and relationship among them.

2.4.3 Evaluation metrics

We used accuracy and diversity to evaluate the recommendation list of APIs. Accuracy is measured in terms of DCG@R for top R recommended APIs. DCG is a popular choice for measuring recommendation accuracy. A higher value of DCG represents better accuracy of the recommendation list. DCG can be defined by the following formula

$$DCG@R = \sum_{i=1}^{R} \frac{2^{r(i)}-1}{log_2(1+i)} \qquad (12)$$

Here i is the position in the recommendation list, and r(i) is the gain or score for the recommended API in the i^{th} position, and $0 \leq r(i) \leq 1$. r(i) value is equal to the normalized popularity value in the mashup category.

We used HMD to measure the diversity of the recommended web APIs. The Hamming distance measures how different the recommended web APIs are in two mashup categories. A bigger hamming distance value implies that the recommendation results are more diverse. HMD can be defined as follow:

$$HMD(m_i, m_j)@R = 1 - \frac{Q(m_i,m_j)}{R} \qquad (13)$$

Here R is the number of web APIs in the recommendation list. Q (m_i, m_j) is the number of same web APIs in the recommendation list of m_i and m_j mashup category. If m_i and m_j contain the same web APIs in the recommendation list, then the HMD (m_i, m_j) = 0 and if there is no such common API in the recommendation list, then HMD (m_i, m_j) = 1.

2.4.4 Baseline Methods

We compared our experimental results with the following methods. Details of these methods are described below:

PopR: This method calculates the number of times each API is used in the mashup category and ranks them based on their count or popularity. It then recommends top R popular API for each mashup category.

KCF: This method applies the K-Means algorithm to cluster mashup services based on their similarity [31]. After this, it applies an item based collaborative filtering (CF) algorithm to recommend top R web APIs for each mashup category [24].

LCF: In this method, LDA is used to cluster mashup services based on its topic feature vector [1] [32]. Item-based CF is applied to recommend top R web APIs for each mashup category.

DL-CF: This method applies DAT-LDA to cluster mashup services based on a description of mashup service and web API and their tags [30]. It then applies Item-based CF to recommend top R web APIs for each mashup category.

ICNC-MF: This is the proposed method in this paper. It applies ICNC to cluster mashup services and then applies Matrix factorization (MF), including popular choices to recommend top R web APIs for each mashup category.

2.4.5 Experimental Results

The below figures show the comparison of experimental results where the number of categories varies from 5 to 20 with step size 5 (i.e. 5/10/15/20). The result shows that our proposed ICNC-MF method achieves a better result than any other baseline method in terms of DCG and HMD for a varying number of recommended web APIs (i.e. R=5/10/20/50). We evaluated our model with varying number of categories and recommended APIs to represent the need for varying number of

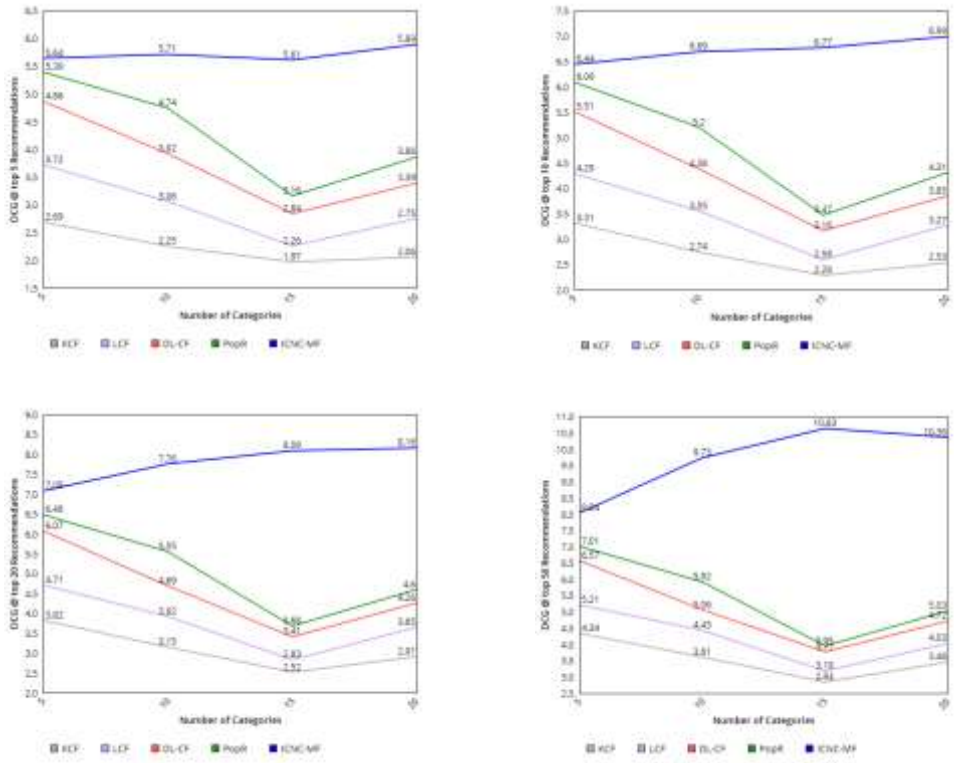

Figure 2. 2 DCG value vs Number of Categories for varying top recommendations

recommended APIs in mashup development for different mashup service categorization. We have the following observations:

ICNC-MF significantly outperforms other baseline methods KCF, LCF, DL-CF, and PopR in terms of recommendation accuracy or DCG value presented in fig. 2.2. Clustering-based recommendation performs better than other baseline methods as it takes into account the relationship among mashup services and draws useful topics achieving more accuracy. PopR is using the same approach for clustering, but MF achieves a better result than PopR as it combines both popular and unpopular APIs.

ICNC-MF also performs better than other baseline methods KCF, LCF, DL-CF and PopR in terms of recommendation diversity or HMD value presented in fig. 2.3. PopR performs worst among the methods as it only recommends the popular APIs without considering the latent

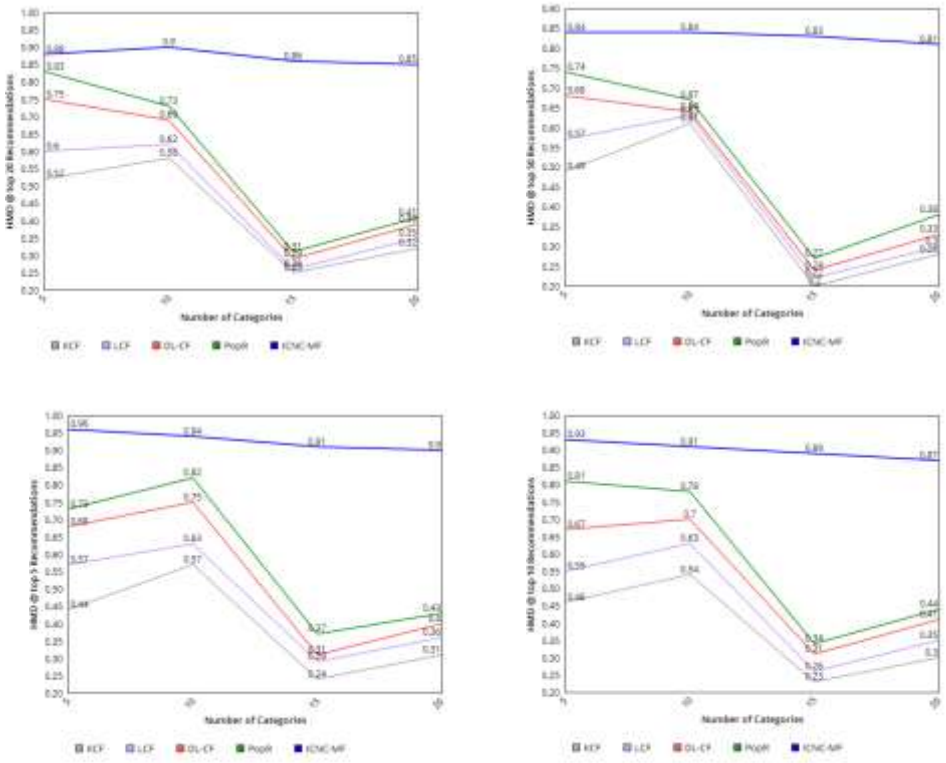

Figure 2. 3 HMD value vs Number of Categories for varying top recommendations

relationship between web APIs and recommending many common ones in each category. DL-CF, LCF, and KCF all apply CF to diversify recommendation results.

Our experimental result shows that ICNC-MF performs better than other baseline methods in terms of accuracy and diversity.

2.5 Related work

2.5.1 Service Recommendation

Service recommendation works can be divided into four categories, mainly: QoS based, functionality based, relationship-based, and hybrid service recommendation method.

Functionality based service recommendation matches the user's requirement and available services and recommends high matching services with the requirement [2, 6-8]. Functionality description of services such as WSDL or Mashup profile files are commonly used to match the

similarity. Li et al. [7] retrieved functional features form the WSDL description of web services using the LDA method. X. Liu et al. [2] proposed to use collaborative topic regression from usage data history and functional description of web services.

Quality of Service (QoS) is an important factor in recommending web services. Z. Zheng et al. [9] presented a method called WsRec, which used the CF method to estimate missing QoS values. Y. Jiang et al. [10] used personalized hybrid CF-based recommendation on top of WsRec and improved the result. K. Fletch et al. [11] proposed a recommendation method that is based on elastic personalized preference over nonfunctional attributes and trade-off on nonfunctional attributes during service selection. X. Chen et al. [12] developed a regional model using the user's physical location to predict QoS values. Tang et al. [13] used both service's location with the user's physical location to estimate missing QoS values. Some researchers have used Matrix factorization in CF to improve QoS value estimation [3, 14-15]. The relationship between services and the used service network has also been considered by several researchers [16-17]. This relationship information among services was retrieved from the invoking, composition, or dependency among them and is used to build a service network. Several researchers worked on the service ecology network [4, 18-19, 25], which uses more service relationship information from service providers, services and users, etc. for better recommendation results.

The hybrid service recommendation method merges the functionality based, QoS based, or relationship-based recommendation services. Kang et al. [20] integrated QoS preference, functional interest, and diversity information to recommend web APIs. Y. Zhong et al. [21] used CF, service evolution, and content matching for service recommendation. L. Yao et al. [23] combined QoS and functionalities using a three-way aspect model proposing a hybrid recommendation technique.

Later they used textual similarity and correlation of APIs to recommend APIs for mashup creation [5].

2.5.2 Clustering-based service recommendation

Clustering and service recommendations are well-known techniques in service recommendation and normally considered as independent processes [26], which may result in poor recommendations result in large and diverse scales. Several researchers have observed the problem and integrated service clustering with service recommendations [26-29]. Y. Zhou et al. [26] presented a heterogeneous service network and modeled services, attributes, and associated entities. Then they performed a random walk integrating link structures and attributes for service clustering and recommendation purpose. D. Skoutas et al. [27] presented dominance relationships between the web services to cluster and web services to rank. J. Zhu et al. [28] presented a landmark-based QoS and clustering-based prediction approaches to recommend web services. Y. Xu et al. [29] presented a collaborative framework of Web service recommendation using clustering-extended matrix factorization.

2.6 Conclusion and Future work

This paper uses an integrated content and network-based service clustering and recommends web APIs using matrix factorization. Service contents (web APIs, description, and tags) are integrated, and the network is developed by building a two-level topic model for mashup clustering. Then we apply a matrix factorization method to recommend web APIs for mashup development. The experimental results show that the matrix factorization based approach achieves better results in terms of accuracy and diversity than the existing baseline methods using the programmableWeb dataset. In the future, we plan to improve clustering accuracy and combine other approaches with

matrix factorization to recommend more accurate and diversified web APIs for mashup development.

2.7 References

[1] B. Xia, Y. Fan, W. Tan, K. Huang, J. Zhang and C. Wu, "Category-Aware API Clustering and Distributed Recommendation for Automatic Mashup Creation," in IEEE Transactions on Services Computing, vol. 8, no. 5, pp. 674-687, Sept.-Oct. 1 2015.

[2] X. Liu and I. Fulia, "Incorporating User, Topic, and Service Related Latent Factors into Web Service Recommendation," 2015 IEEE International Conference on Web Services, New York, NY, 2015, pp. 185-192.

[3] Z. Zheng, H. Ma, M. R. Lyu and I. King, "Collaborative Web Service QoS Prediction via Neighborhood Integrated Matrix Factorization," in IEEE Transactions on Services Computing, vol. 6, no. 3, pp. 289-299, July-Sept. 2013.

[4] W. Xu, J. Cao, L. Hu, J. Wang and M. Li, "A Social-Aware Service Recommendation Approach for Mashup Creation," 2013 IEEE 20th International Conference on Web Services, Santa Clara, CA, 2013, pp. 107-114.

[5] L. Yao, X. Wang, Q. Z. Sheng, W. Ruan and W. Zhang, "Service Recommendation for Mashup Composition with Implicit Correlation Regularization," 2015 IEEE International Conference on Web Services, New York, NY, 2015, pp. 217-224.

[6] L. Liu, F. Lecue, and N. Mehandjiev. “Semantic content-based recommendation of software services using context.” ACM Transactions on the Web, 7(3): 17-20, 2013.

[7] C. Li, R. Zhang, J. Huai, X. Guo and H. Sun, "A Probabilistic Approach for Web Service Discovery," 2013 IEEE International Conference on Services Computing, Santa Clara, CA, 2013, pp. 49-56.

[8] S. Meng, W. Dou, X. Zhang and J. Chen, "KASR: A Keyword-Aware Service Recommendation Method on MapReduce for Big Data Applications," in IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 12, pp. 3221-3231, Dec. 2014.

[9] Z. Zheng, H. Ma, M. R. Lyu and I. King, "QoS-Aware Web Service Recommendation by Collaborative Filtering," in IEEE Transactions on Services Computing, vol. 4, no. 2, pp. 140-152, April-June 2011.

[10] Y. Jiang, J. Liu, M. Tang and X. Liu, "An Effective Web Service Recommendation Method Based on Personalized Collaborative Filtering," 2011 IEEE International Conference on Web Services, Washington, DC, 2011, pp. 211-218.

[11] K. K. Fletcher, X. F. Liu, and M. Tang. “Elastic Personalized Nonfunctional Attribute Preference and Trade-off based Service Selection.” ACM Transactions on the Web, 9(1): 1-26, 2015.

[12] X. Chen, Z. Zheng, Q. Yu and M. R. Lyu, "Web Service Recommendation via Exploiting Location and QoS Information," in IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 7, pp. 1913-1924, July 2014.

[13] M. Tang, Y. Jiang, J. Liu and X. Liu, "Location-Aware Collaborative Filtering for QoS-Based Service Recommendation," 2012 IEEE 19th International Conference on Web Services, Honolulu, HI, 2012, pp. 202-209.

[14] W. Lo, J. Yin, S. Deng, Y. Li and Z. Wu, "Collaborative Web Service QoS Prediction with Location-Based Regularization," 2012 IEEE 19th International Conference on Web Services, Honolulu, HI, 2012, pp. 464-471.

[15] P. He, J. Zhu, Z. Zheng, J. Xu and M. R. Lyu, "Location-Based Hierarchical Matrix Factorization for Web Service Recommendation," 2014 IEEE International Conference on Web Services, Anchorage, AK, 2014, pp. 297-304.

[16] C. Zhao, C. Ma, J. Zhang, J. Zhang, L. Yi and X. Mao, "HyperService: Linking and Exploring Services on the Web," 2010 IEEE International Conference on Web Services, Miami, FL, 2010, pp. 17-24.

[17] Q. Wu, A. Iyengar, R. Subramanian, I. Rouvellou, I. Silva-Lepe, and T. A. Mikalsen. “Combining quality of service and social information for ranking services.” Service-Oriented Computing. Springer Berlin Heidelberg, 2009, pp. 561–575.

[18] E. Wittern, J. Laredo, M. Vukovic, V. Muthusamy and A. Slominski, "A Graph-Based Data Model for API Ecosystem Insights," 2014 IEEE International Conference on Web Services, Anchorage, AK, 2014, pp. 41-48.

[19] K. Huang, Y. Fan and W. Tan, "An Empirical Study of Programmable Web: A Network Analysis on a Service-Mashup System," 2012 IEEE 19th International Conference on Web Services, Honolulu, HI, 2012, pp. 552-559.

[20] G. Kang, M. Tang, J. Liu, X. (. Liu and B. Cao, "Diversifying Web Service Recommendation Results via Exploring Service Usage History," in IEEE Transactions on Services Computing, vol. 9, no. 4, pp. 566-579, July-Aug. 1 2016.

[21] Y. Zhong, Y. Fan, K. Huang, W. Tan and J. Zhang, "Time-Aware Service Recommendation for Mashup Creation in an Evolving Service Ecosystem," 2014 IEEE International Conference on Web Services, Anchorage, AK, 2014, pp. 25-32.

[22] C. Li, R. Zhang, J. Huai and H. Sun, "A Novel Approach for API Recommendation in Mashup Development," 2014 IEEE International Conference on Web Services, Anchorage, AK, 2014, pp. 289-296.

[23] L. Yao, Q. Z. Sheng, A. H. H. Ngu, J. Yu and A. Segev, "Unified Collaborative and Content-Based Web Service Recommendation," in IEEE Transactions on Services Computing, vol. 8, no. 3, pp. 453-466, May-June 1 2015.

[24] W. Gao, L. Chen, J. Wu and H. Gao, "Manifold-Learning Based API Recommendation for Mashup Creation," 2015 IEEE International Conference on Web Services, New York, NY, 2015, pp. 432-439.

[25] K. Huang, Y. Fan, W. Tan and X. Li, "Service Recommendation in an Evolving Ecosystem: A Link Prediction Approach," 2013 IEEE 20th International Conference on Web Services, Santa Clara, CA, 2013, pp. 507-514.

[26] Y. Zhou, L. Liu, C. S. Perng, A. Sailer, I. Silva-Lepe and Z. Su, "Ranking Services by Service Network Structure and Service Attributes," 2013 IEEE 20th International Conference on Web Services, Santa Clara, CA, 2013, pp. 26-33.

[27] D. Skoutas, D. Sacharidis, A. Simitsis and T. Sellis, "Ranking and Clustering Web Services Using Multicriteria Dominance Relationships," in IEEE Transactions on Services Computing, vol. 3, no. 3, pp. 163-177, July-Sept. 2010.

[28] J. Zhu, Y. Kang, Z. Zheng and M. R. Lyu, "A Clustering-Based QoS Prediction Approach for Web Service Recommendation," 2012 IEEE 15th International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops, Shenzhen, Guangdong, 2012, pp. 93-98.

[29] Y. Xu, J. Yin, Y. Li. “A collaborative framework of web service recommendation with clustering-extended matrix factorisation.” in International Journal of Web and Grid Services, vol. 12, Issue 1, pp. 1-25, 2016.

[30] B. Cao, X. Liu, J. Liu, and M.Tang. “Effective Mashup Service Clustering Method by Exploiting LDA Topic Model from Multiple Data Sources.” Asia-Pacific Services Computing Conference. Springer International Publishing, 2015, pp. 165-180.

[31] L. Chen, L. Hu, J. Wu, Z. Zheng, J. Ying, Y. Li, and S. Deng. “WTcluster: utilizing tags for web service clustering.” International Conference on Service-Oriented Computing. Springer Berlin Heidelberg, 2011, pp. 204-218.

[32] L. Chen, Y. Wang, Q. Yu, Z. Zheng, and J. Wu. “WT-LDA: User Tagging Augmented LDA for Web Service Clustering.” International Conference on Service-Oriented Computing. Springer Berlin Heidelberg, 2013, pp. 162-176.

[33] B. Cao et al., "Mashup Service Clustering Based on an Integration of Service Content and Network via Exploiting a Two-Level Topic Model," 2016 IEEE International Conference on Web Services (ICWS), San Francisco, CA, 2016, pp. 212-219.

[34] R. Torres, B. Tapia and H. Astudillo, "Improving Web API Discovery by Leveraging Social Information," 2011 IEEE International Conference on Web Services, Washington, DC, 2011, pp. 744-745.

[35] H. Elmeleegy, A. Ivan, R. Akkiraju and R. Goodwin, "Mashup Advisor: A Recommendation Tool for Mashup Development," 2008 IEEE International Conference on Web Services, Beijing, 2008, pp. 337-344.

[36] S. R. Chowdhury, F. Daniel, and F. Casati, “Efficient, interactive recommendation of mashup composition knowledge,” in International Conference on Service-Oriented Computing. Springer Berlin Heidelberg, 2011, pp. 374–388.

Chapter 3: Integrated Topic Modeling and User Interaction Enhanced Web-API Recommendation Model for Mashup Application Development using Regularized Matrix Factorization Method

3.1 Abstract

Mashup application developers combine relevant web APIs from existing sources. Still, developers often face challenges in finding appropriate web APIs as they have to go through thousands of available ones. Recommending relevant web APIs might help, but very low API invocation from mashup applications creates a sparse dataset for the recommendation models to learn about the mashups and their invocation pattern, ultimately affecting their accuracy. Effectively reducing sparsity and using supplemental information such as mashup and web API specific features that trigger mashups to invoke the same web APIs in their applications and web APIs to be used together by a mashup can help to generate more accurate and useful recommendations. In this work, we developed a novel web API recommendation model for mashup application, which uses two-level topic modeling of mashups and user interaction with mashup and web APIs sequentially to reduce the sparsity of the initial dataset. Then, we applied regularized matrix factorization with the mashup and web API embeddings. These embeddings integrate 'mashup to mashup' and 'web API to web API' relationships with 'mashup to web API' invocation analysis. Compared with existing web API recommendation models, our model achieved 54% more precision, 36.4% more Normalized Discounted Cumulative Gain (NDCG), and 36% more recall value over other baseline models on a dataset collected from programmbleWeb.

3.2 Introduction

Mashup application development is a widespread software development practice where developers usually integrate web APIs from different sources to create a new streamlined service for end-users. As an example, Trendsmap[1] is a map mashup service, which incorporates the twitter trends and google map web services to provide a mapping of trending topics locally and globally in real-time. Mashup applications help with situational software development where a software application needs to be developed for only a small number of users satisfying their needs [1]. Due to the limited scope of these software applications, the only way these software applications would be cost-effective if they can be developed in a short period yielding a low development cost [1]. Mashup development encourages end-user software development. End users are often domain experts and with little programming knowledge. They develop applications collaboratively from existing sources rather than going through the lengthy software development cycle.

Due to recent growth in publishing web APIs by different web and business organizations, there are thousands of APIs available nowadays. ProgrammableWeb[2] is a popular online repository for mashup and web APIs. As of May 2020, programmableWeb[1] contains 22,992 APIs, and on average, 2019 new APIs are being added yearly on their site [2]. This massive number of APIs is causing a challenging issue for mashup application developers, which is choosing the relevant APIs from these thousands of available ones. It is not a feasible task for a developer to go through all of the APIs and select the right ones for their corresponding application. Instead, if we can recommend relevant APIs to the mashup application developers, it would be easier for them to find the appropriate APIs for their applications.

1. https://www.trendsmap.com/
2. https://www.programmableweb.com/

Different research work focuses on this problem and developed various models and algorithms to recommend relevant web APIs to the mashup application developers. Most of these works focus on the functional and non-functional properties of web APIs and user requirements in the recommendation process. On the non-functional approaches, web APIs' Quality of Services (QoS), locations, etc. are considered in the recommendation [3, 4, 5]. On the functional approaches, the popularity of web-APIs, co-invocation pattern of web APIs, topic modeling, learning-based models, different kinds of information such as description, tags, category, etc. are considered in different web API recommendation models [6, 7, 8].

Most of these approaches use the web API invocation from the mashup applications to learn about the mashups and the kind of APIs it usually invokes, which they later use to recommend new web APIs to the mashup applications. However, low API invocation from mashups is a typical challenging issue of these models. On average, a mashup application only invokes three to five web-APIs, but there are thousands of web-APIs available for recommendation [9]. This low invocation generates a very sparse dataset for recommendation models to learn about individual mashup applications and the general pattern, hidden factors, and relations among mashups and web APIs, which are necessary to generate useful recommendations. As a result, the web API recommendations made by these models would not be accurate and useful for mashup applications.

One common way to minimize the impact of this issue is by reducing the sparsity of the initial mashup-web API invocation dataset. In the mashup domain, the clustering-based approach is one of the effective ways to generate a denser dataset. It identifies similar mashup services and uses web API invocation data from these services to reduce the sparsity of the mashup-web API invocation data. In the clustering process, different attributes of mashups' and web APIs' such as invocation history, quality of services of APIs, etc. are usually used to identify similar mashups,

which is later in the recommendation [9, 10, 11, 12, 13]. However, clustering mashups is not sufficient enough for reasonably accurate web API recommendation for mashup applications. This scenario is evident in the experiments of [14], where clustering did not generate reasonable accuracy on newer and larger mashup-web API datasets.

User interaction data with mashup and web APIs can be another source of data to reduce sparsity in the mashup - web API recommendation process. Incorporating user interaction data into the web API recommendation process has shown some promising results in the user-based personalized web API recommendation [14] [15]. These models recommend APIs for users analyzing their interactions with mashups and web APIs, but they do not recommend APIs for a particular mashup application. However, user-specific interaction data may help the general web API recommendation for mashup applications too. If many users interact with two APIs, then there should be some API specific features between these two APIs, which is influencing users' interaction also. These web API specific features and factors can improve the web API recommendation for mashup application. A thorough experimental investigation is required to validate this concept of whether existing user-interaction data with mashup and web APIs can improve the web API recommendation for mashup application too.

Another way we can minimize the impact of low web API invocation by using the additional information in the recommendation process. From our analysis, many mashups invoke the same APIs in their applications, and many have overlapping APIs in their invocation history. This 'mashup to mashup' information can help us to find out specific factors by which these mashups are related to each other in their invocation pattern of APIs. Also, we analyzed that there are many web APIs that are generally used together by different mashups. Similarly, 'web API to web API' information can help us to find out specific factors by which the web APIs are invoked or not

invoked together by mashup applications. Integrating 'mashup to mashup' and 'web API to web API' specific analyses with the traditional 'mashup to web API' invocation analysis can help the web API recommendation processes to achieve higher accuracy and generate useable recommendations for the mashup applications.

Considering the above insights, we developed a web API recommendation model for mashup applications. In our developed model, we applied two techniques sequentially to identify similar and related mashups and web APIs. The first technique uses Integrated Content and Network-based mashup Clustering (ICNC), which is developed by [9] to group similar mashup services together. The second technique applies user interaction data to identify related mashups and web APIs. We used these similar and related mashups and web APIs to reduce the sparsity in the mashup-web API invocation dataset. On the condensed dataset, we then applied Regularized Matrix Factorization with the user and item Embedding (RME). RME model is developed by [16] for general recommendation purposes, which can use additional information as embeddings. We adapted the RME model so that it can operate with mashup and web API embedding as supplemental information with the web API invocations. With explicit embedding, this model identified the latent factors associated with 'mashup to mashup' similar API invocation, and 'web API to web API' invoked/not invoked by the same mashups and integrated them with the analysis of traditional mashup to web API invocation. Using these steps, our model generated the top N web API recommendation for mashup application. To our knowledge, our model is the first approach, which exploited these embeddings to identify the mashup and web API specific latent features in the web API recommendation model. Also, our model is the first approach, which used the users' interaction data to identify related web APIs and mashups and later used it for web API recommendation for mashup applications.

Let's go through an example to understand how our model recommends web APIs to a mashup. Let's assume; we want to recommend web APIs for mashup 'Music Updated[1]'. Music Updated only invoked Youtube[2] and Last.fm[3] web APIs. Our model will find related mashups and APIs using ICNC [9] and user interaction data. Let's assume, related mashups of 'Music Updated' invoke Feed.fm[4] API and related APIs of the invoked APIs are SoundCloud[5], Spotify[6] APIs. Our model will learn about 'Music Updated' mashup using YouTube, SoundCloud, Spotify, Last.fm, and Feed.fm APIs, instead of just using YouTube, Last.fm APIs. Our model will identify from mashup specific knowledge that related mashups generally use APIs that provide the functionality to access and play radio content. From web API specific knowledge, it will identify that music-related APIs are usually used together by a mashup. After combining this information with actual invocation data, our model will recommend Feed.fm, shufflerfm[7], SoundCloud, Spotify, Grooveshark[8] APIs, which are a combination of music and radio-related APIs.

In this paper, we make the following contributions:

1. We propose a new web API recommendation model for mashup application development exploiting mashup clustering, user interaction, and mashup and web API embedding via RME.

2. Our experimental results on a real-life mashup-web API-user dataset collected from programmableWeb showed that our model achieved 54% more precision, 36.4% more Normalized Discounted Cumulative Gain (NDCG), and 36% more recall value over other models.

1. https://www.programmableweb.com/mashup/music-updated
2. https://www.programmableweb.com/api/youtube-rest-api
3. https://www.programmableweb.com/api/lastfm-rest-api-v20
4. https://www.programmableweb.com/api/feedfm-rest-api
5. https://www.programmableweb.com/api/soundcloud-rest-api
6. https://www.programmableweb.com/api/spotify-web-rest-api-v10
7. https://www.programmableweb.com/api/shufflerfm-rest-api
8. https://www.programmableweb.com/api/grooveshark-rest-api

The rest of the paper is organized in the following way. We discuss related work in the field of web API recommendation for mashup development, and then we describe our proposed model and experimental results and evaluation of our work.

3.3 Related Work

Most of the current research work in web API recommendation focuses on Quality of Services (QoS), web API functionality, and latent/implicit relationship among APIs/mashups. QoS based service recommendation approaches focuses on user's preference on non-functional attributes of APIs, estimates the missing QoS values using different recommendation algorithms such as collaborative filtering (CF), matrix factorization (MF) to recommend APIs with matching requirement [3, 4, 5]. Functionality based service recommendations usually take user's requirements as input and identify key topics and attributes in the requirement, web APIs, and mashups using different topic modeling. These models typically go through the mashup/web API document to figure out the key topics from both the content and the user requirement [6, 7, 8] and recommend APIs with matching topics. Implicit relationship-based service recommendations usually use the co-vocation of web APIs in the mashups, dependency among items, the network of mashup services, shared tags/categories, etc. are exploited to identify the latent relationships among web APIs and mashup services. Service ecology network considers information from service providers, users about service relationships on top of existing service information in the web API recommendation process [17, 18].

Due to the sparse nature of the mashup-web API dataset, clustering approaches are also popular in the web-service recommendation process. Usually, the clustering process is performed independently, and the clustering result is integrated back into the service recommendation part later. For the clustering process, [19] considered different attributes, and entities of services

modeling a divergent service network and applied a random walk based method to integrate attributes and structures of links. The dominance relationship between services [20] is used in the clustering mechanism of services and later in the ranking process of web services. Service clustering is also used with a landmark-based QoS prediction in the recommendation process of web services. Clustering extended matrix factorization is also used by [21] in web service recommendation. However, none of these clustering-based approaches user interaction data with the clustering results to create a denser dataset.

The matrix factorization (MF) method is recently gaining a lot of popularity in the service recommendation area. Researchers mostly vary their use of additional information on the regularization process in MF based models. [22] developed used a trace norm regularization in MF to recommend web services with the best QoS values. Additional information such as location information considering the IP address, trust propagation, time-sensitive modeling technique, data weighting approach, are also infused with matrix factorization model [23, 24, 25] for web service recommendations.

Deep learning based approaches have been used in the API recommendations for the mashup applications. [27] utilized the Convolutional Neural Network (CNN), and Long Short Term Memory (LSTM) technologies with the natural language processing technologies to recommend APIs. We implemented CNN, and LSTM based models and compared the performance with our developed ExICNC-RME model.

3.4 Web-API Recommendation Model For Mashup Development

In this section, we discussed our developed web-API recommendation model for mashup application development. This section is divided into two sub-sections. In the first sub-section,

we explained how we employed two different techniques to generate a denser mashup-web API dataset from its initial sparse form. In the second sub-section, we discussed how we applied the RME with mashup and web API embedding on the condensed dataset for the recommendation process.

3.4.1 Denser Mashup - WebAPI Dataset Generation:

This section describes the clustering and user interaction based techniques we used to reduce the sparsity on the original mashup - web API dataset:

1) Integrated Content and Network-based Clustering (ICNC) for Denser Dataset:

We used Integrated Content and Network-based Clustering (ICNC) to group similar mashup services together. Then we used the grouping information to reduce the sparsity of the original mashup – web API dataset. The ICNC [9] method and its incorporation are explained in the following two subsections.

a) ICNC Method

The ICNC [9] model builds a mashup service document for each mashup, which contains the category, name, description, composing web APIs, and associated tag information. Then, it builds a mashup service network based on the similar web-APIs invocation and shared tag information. For each mashup service document, the topic distribution is generated from its content using the Latent Dirichlet Allocation (LDA) model. Then, the topic distribution of the mashup services belonging in the same network is incorporated into the topic distribution of that mashup service based on the concept that useful and signature topics of mashups are distributed both at the mashup's own content and over the network of that mashup. Topic distribution is used to calculate similarity among mashup services and later identify groups of similar mashup services.

b) Sparsity Reduction in the Mashup-WebAPI dataset using ICNC Method:

Using the ICNC [9] method, we obtained different clusters of mashup services, where each cluster contained similar mashup services together based on their content and network information. Similar mashup services should invoke similar and related web APIs. We used this information to generate a condensed mashup - web API matrix with less sparsity. The basic idea of this approach is that if two mashups belong to the same cluster, then if one mashup invokes an API, the other mashups in the group will invoke that API too. We generated a web API set from all invoked web APIs by the mashups, then all the web APIs from this set will be invoked by all the mashups in a group. In this way, all mashups in a group will share the same web API invocation history. As an example, a mashup group contains two mashups SendMusic2.Me[1] and Viral Music List[2]. SendMusic2.Me invokes Youtube and Facebook[3] API, whereas Viral Music List invokes Facebook and Spotify API. Group web API set will contain three web APIs {Youtube, Facebook, Spotify}. After this denser mashup-web API matrix generation, SendMusic2.Me and Viral Music List will have Youtube, Facebook, Spotify APIs in their invocation history. This process can be shown in the following algorithm:

APISet = Empty List {}

for each mashup m_x in $cluster_a$

for each API_i in m_x-APIs

if API_i is not in APISet

Add API_i into APISet

1: https://www.programmableweb.com/mashup/sendmusic2.me
2: https://www.programmableweb.com/mashup/viral-music-list
3. https://www.programmableweb.com/api/facebook

for each mashup m_x in cluster$_a$

for each API$_i$ in APISet

if API$_i$ is not in m_x-APIs

Add API$_i$ in m_x-APIs

Here, *APISet* is the set of web APIs for a cluster of mashups, and *cluster$_a$* is a cluster of similar mashups. m_x is a mashup in a cluster, and *API$_i$* is a web API. *m_x-APIs* is the list of web APIs which are invoked by mashup m_x. This process will add new APIs into the invoked API list of mashups, which will reduce the sparsity of the dataset.

2) User Interaction with Mashups and Web-APIs for Sparsity Reduction in the Dataset

We used user interaction data with the mashup and web APIs to reduce sparsity in the mashup-web API dataset further. We applied the following two techniques to condense the dataset.

a) User Interaction with Web-APIs:

Users can create and follow APIs in the dataset we used. If a user creates or follows multiple APIs, then these APIs are related to each other that captured the user's attention. On the condensed dataset from the ICNC method, we used this correlation information from the create/follow information to condense the dataset further. We identified the related API pairs from user interaction with web APIs, used those pairs to condense the dataset. The idea is that if a mashup invokes an API from a related API pair, the mashup will invoke the other API in that pair too. We will show the entire process with the following example with actual API names for better understandability:

Let, user x follows some APIs and creates some APIs. First, we combined all the APIs that user x either followed or created. F, C, and L contain the APIs that user x followed, created, and the combined APIs respectively:

F = [SoundCloud, Spotify, YouTube]

C = [TouchTunes Jukebox[1], Soundtrap[2]]

L = [SoundCloud, Spotify, YouTube, TouchTunes Jukebox, Soundtrap]

From this combined list, L, we generated an all-possible combination of API pairs. This list of API pairs is the related API list (R_x) from user x:

R_x = [(SoundCloud, Spotify), (Spotify, YouTube), (SoundCloud, YouTube), ……..(YouTube, TouchTunes Jukebox), (TouchTunes Jukebox, Soundtrap)]

In this way, we created the related API list for every user in our dataset. Finally, we combined the related API list from all users to get a global related API list, R that contains all pairs of related API lists from all users.

R = [(SoundCloud, Spotify), (Spotify, YouTube), …….(Google Photos[3], Flickr[4]), (Weatherbit Severe Weather Alerts[5], National Weather Service[6])]

We used this global related API list R to condense the mashup-web API matrix. For a related API pair (API_i, API_j), if mashup x invokes API_i, then in the condensed mashup-web API matrix, mashup x will invoke API_j too. For each API pair, we will browse through the mashups to

1. https://www.programmableweb.com/api/touchtunes-jukebox-rest-api
2. https://www.programmableweb.com/api/soundtrap-rest-api
3. https://www.programmableweb.com/api/google-photos-rest-api-v10
4. https://www.programmableweb.com/api/flickr-rest-api
5. https://www.programmableweb.com/api/weatherbit-severe-weather-alerts-rest-api-v20
6. https://www.programmableweb.com/api/national-weather-service-nws-rest-api

propagate the co-relationship information of the APIs. As an example: Music Updated mashup uses Youtube API, it will also use SoundCloud, Spotify Web, TouchTunes Jukebox, Soundtrap in the condensed version, as each of these APIs is related with Youtube. The following algorithm summarizes this procedure:

for each Mashup m_x

for each API pair (API_a, API_b) *in R*

if API_a *is in* m_x*-APIs*

if API_b *is not in* m_x*-APIs*

Add API_b *in* m_x*-APIs*

if API_b *is in* m_x*-APIs*

if API_a *is not in* m_x*-APIs*

Add API_a *in* m_x*-APIs*

Here, API_a and API_b are particular web APIs. (API_a, API_b) is an API pair in the global related API list, R and m_x is a mashup service. m_x*-APIs* is the list of web APIs which are invoked by mashup m_x. This process will add new APIs into the invoked API list of mashups, which will reduce the sparsity of the dataset.

b) User Interaction with Mashups:

Users can create and follow mashups in our dataset too. We used this user-mashup interaction information to condense the mashup-web API dataset also. From mashup users' created/followed mashup list, we generated a list of mashup pairs, which are related to each other. Then, we combined the related mashup pairs from all users to get the global related mashup pairs list from

all users. This process is similar to global related API pair list generation, as described in the previous section. Using the global mashup pairs list, we condense the mashup-web API matrix. If two mashups are related, then if a mashup invokes an API, the other mashup will invoke that too in the condense mashup-web API matrix. As an example: user x follows mashup SoundYouNeed[1] and Music Updated. SoundYouNeed invokes YouTube, SoundCloud, Spotify, Grooveshark API. Music Updated uses YouTube, Last.fm API. After this condense process, SoundYouNeed and Music Updated will both invoke YouTube, SoundCloud, Spotify, Grooveshark, Last.fm APIs. This process is summarized in the following algorithm:

for each mashup pair (m_x, m_y) in S

for each API_i in m_x-APIs

if API_i is not in m_y-APIs

Add API_i in m_y-APIs

for each API_i in m_y-APIs

if API_i is not in m_x-APIs

Add API_i in m_x-APIs

Here, m_x, m_y is two mashup services. (m_x, m_y) is a mashup pair in the global related mashup list, S. API_i is a particular web API. m_x-*APIs* and m_y-*APIs* is the list of web APIs which are invoked by mashup m_x and $m_{y,}$ respectively.

These three processes will also add new APIs into the invoked API list of mashups. As we added new APIs into the invoked list, the mashup – web API matrix will have reduced sparsity

1. https://www.programmableweb.com/mashup/soundyouneed

than created from the original dataset. On the final condensed matrix, we applied the RME [16] model with mashup and web API embeddings. Before we describe the RME model, we will give a brief overview of the matrix factorization method.

3.4.2 Matrix Factorization

Matrix Factorization identifies the latent factors associated with the user and item in the dataset. Given, a dataset of user ratings on items, R with |U|× |I| dimension, where U is the set of users, and I is the set of items. Then it factorizes the R matrix into two matrices P and Q for K latent features related to user and item matrices. The size of P and Q is |U| × |K| and |V| × |K|, and they represent the latent features associated with user and items. Each row in P represents the association between a user and the latent user features. Each row in Q represents the association between an item and the latent item features. The purpose of these two matrices is to approximate the initial matrix R using the dot product of user and item latent features in the following way:

$$R \approx P \times Q^T = \hat{R}$$

With L2-regularization, it minimizes the following error function:

$$e_{ij}^2 = (r_{ij} - \sum_{k=1}^{K} p_{ik} q_{kj})^2 + \frac{\beta}{2} \sum_{k=1}^{K} (||P||^2 + ||Q||^2)$$

Here, β is a regularization parameter, r_{ij} is an actual rating for user i and item j. p_{ik} and q_{kj} are the latent vectors for user and item, respectively.

3.4.3 Regularized Matrix Factorization with Embedding (RME) Model:

RME [27] is a matrix factorization based model with the user and item embedding. It applies the word-embedding techniques from natural language processing to generate the user and item embedding. The basic idea is that for a particular user if we sort the liked items in a specific order

based on their timestamp value, it can be used as a sequence to learn about latent features associated with user and items. We will use the condensed mashup-web API matrix and then apply the RME [27] model to predict the missing web API invocation values. The RME [27] model enables to incorporate user and item specific supplemental information as embeddings with the original user-item data. We used the following three mashup and web API specific embeddings:

1) Mashup to Mashup Embedding:

This embedding analyzes the mashup specific features that cause mashups to invoke similar APIs in their applications. A mashup to mashup co-invocation matrix is generated with |mashup| × |mashup| dimension, and if a web API is invoked by both the mashups, then the associated value is enabled. An MF model is applied to this matrix to identify the mashup specific latent factors.

2) Co-invoked web API to web API Embedding:

This embedding analyzes the web API specific features that cause web APIs to be invoked by the same mashup. A web API to web API co-invoked matrix is generated with |web API| × |web API| dimension, and if two web APIs are used by the same mashups, then the associated value is enabled. An MF model on this matrix is applied to identify the web API specific latent factors.

3) Co-uninvoked web API to web API Embedding:

This embedding identifies specific features that cause web APIs to be not invoked by the same mashup. In web API to web API co-uninvoked matrix, if two web APIs are not invoked by the same mashups, then the associated value is enabled.

With these three matrices, RME model implements a joint learning model combining weighted Mashup to Web API Invocation (MWAI), Invoked Web API Embedding (IWAE), Uninvoked

Web API embedding (UWAE), and Mashup Embedding (ME) and minimizes the following objective function:

$$\mathcal{L} = \frac{1}{2}\sum_{u,p} w_{up}\left(M_{up} - \alpha_U^T \beta_p\right)^2 \ (MWAI)$$

$$+ \frac{1}{2}\sum_{X_{pi}\neq 0} w_{pi}^{(+p)}\left(X_{pi} - \beta_p^T \gamma_i - b_p - c_i\right)^2 \ (IWAE)$$

$$+ \frac{1}{2}\sum_{Y_{pi'}\neq 0} w_{pi'}^{(-p)}\left(Y_{pi'} - \beta_p^T \delta_{i'} - d_p - e_{i'}\right)^2 \ (UWAE)$$

$$+ \frac{1}{2}\sum_{Z_{uj}\neq 0} w_{uj}^{(u)}\left(Z_{uj} - \alpha_u^T \theta_j - f_u - g_j\right)^2 \ (ME)$$

$$+ \frac{1}{2}\lambda\left(\sum_u ||\alpha_u||^2 + \sum_p ||\beta_p||^2 + \sum_i ||\gamma_i||^2 + \sum_{i'} ||\delta_{i'}||^2 + \sum_j ||\theta_j||^2\right)$$

Here, M is the mashup-web API matrix, U is the latent factor matrix for mashups, P is the latent factor matrix for web APIs, X, Y, and Z are invoked, uninvoked web APIs and mashup co-invocation matrix respectively. α_u, β_p, γ_i, θ_j are latent factor vector of mashup u, web API p, co-invoked web API context, co-uninvoked web API context, and mashup context respectively. b, d, c, e, f, g are co-invoked web APIs, co-uninvoked web APIs, co-invoked web APIs context, co-uninvoked web APIs context, mashup and mashup bias, respectively.

We predicted the invocation values for the missing web APIs applying the RME model on the condensed dataset. We sorted the predicted values and generated the top N recommendations for each mashup application.

3.5 Experiment

In this section, we described the experiment we conducted and the associated results we observed.

3.5.1 Experimental Dataset

We used the web-API dataset provided by [14] in our experiment. This dataset was crawled from programmbleWeb, which is one of the popular online repositories for web APIs and mashups. This dataset contained the mashup-API invocation and user interaction history. In the user-to-user interaction history, there is a list of users with their followed user, mashups, and APIs lists. For each mashup, we had a short description, URL, associated tags, invoked APIs. For each API, we had API name, associated URL, tags, primary category, secondary categories, a short description of the API in the dataset. In summary, there were 17,564 web-APIs, 6270 mashups, 87, 857 user profiles in the dataset. For more details on the dataset, please refer to [14].

3.5.2 Evaluation Metrics:

To compare the performance of our developed web API recommendation model with other baseline models, we used Normalized Discounted Cumulative Gain (NDCG), Recall, Mean Average Precision (MAP) evaluation matrices. These evaluation metrics measure the recommendation quality and the accuracy and coverage of the recommended APIs.

Normalized Discounted Cumulative Gain (NDCG): NDCG is a measure of recommendation ranking quality. It calculates the Discounted Cumulative Gain (DCG) value in the following way:

$$DCG_N = \sum_{i=1}^{N} \frac{rel_i}{log_2(i+1)}$$

Here, i is the position in the recommendation, and rel_i is the relevance score of the recommended item at position i. N is the total number of recommendations made. This Discounted Cumulative Gain (DCG) value is normalized with the Ideal DCG (IDCG) value measured from the test dataset. NDCG is calculated using the following equation:

$$NDCG = \frac{DCG}{IDCG}$$

A higher NDCG value represents a more top quality recommendation, and the maximum value for NDCG is 1.0, which constitutes an ideal recommendation.

Recall: Recall value captures the percentage of relevant items that are captured by the top-N recommendation. Recall value is calculated using the following equation:

$$Recall = \frac{True\ Positives}{True\ Positives\ +\ False\ Negatives}$$

Like NDCG, a higher recall value represents a better recommendation result with a maximum amount of 1.0.

Mean Average Precision (MAP): MAP measures the average precision result for each iteration in the experiment. The precision value measures the percentage of recommended items are relevant using the following equation:

$$Precision\ = \frac{True\ Positives}{True\ Positives\ +\ False\ Positives}$$

3.5.3 Baseline Models:

ICNC-CF: The method was developed by [9]. This method uses the ICNC [9] method to cluster the related mashup services together and then collaborative filtering (CF) model to predict the missing values, which is later used to generate top N recommendation.

ICNC-MF: This method was developed by [26]. This method uses the matrix factorization with L2-regularization for recommendation after applying the ICNC [9] method.

ICNC-RME: This method ICNC method to reduce sparsity on the mashup – web API dataset and then applies the RME model with the mashup-mashup co-occurrence matrix, co-liked web API matrix, and co-disliked web API matrix.

Mashup-CF: This method applies the collaborative filtering (CF) method on the initial mashup-web API matrix.

Mashup-MF: This method applies the matrix factorization method with L2-regularization on the initial mashup-web API matrix.

Mashup-RME: This method uses the RME on the initial mashup-web API matrix.

ExMashup-RME: This method uses the user interaction data on top of the initial mashup-web API data to reduce sparsity and then apply the RME model. This method is similar to the RMFUP model developed by [9]. However, the RMFUP model recommends API to the user, not to mashup application. It also takes the user requirement to filter the recommendation result, and this is not the case here. In sparsity reduction, RMFUP filtered some APIs based on functional information of the APIs such as Language and Data formats supported, etc. Functional information is not available on the publicly available data, and we did not filter any API.

LSTM-Rec: We implemented this LSTM [27] based recommendation model. In this model, we provided the initial mashup and web APIs as input from the initial dataset, and then applied this LSTM based mode to generate API recommendations for the mashups. The internal architecture, and other details is described in section 1.13.5.

CNN-Rec: This is a CNN [27] based recommendation model. We fed the initial mashup to API dataset to this model in order to generate API recommendations for the mashups. Please refer to section 1.13.5 for the internal architecture, and other details is described in.

NCF-Rec: This is a neural network based collaborative filtering model [28]. We utilized the initial mashup to web API dataset in this mode to generate the final recommendation results. The internal architecture, and other details is described in section 3.5.5.

ExICNC-RME: This is the proposed model in this paper. On the initial dataset, this model uses the ICNC method and user interaction data to reduce sparsity in the dataset. Then it applies the RME model with mashup and web API embedding for the recommendation purpose.

3.5.4 Experimental Setup

For each model, we evaluated the model's performance at different top N recommendations. At each top N recommendation, we performed cross-validation with five-folds, two-repetitions, and averaged the results. In each iteration, from the overall mashup - web API invocation dataset, we identified K test mashups (20 percent of all mashups), and N web APIs for each of the mashups from their associated invocation history. This K mashups, and their associated N APIs was our ground truth model. For each of the test mashups, we removed their associated API invocations, which are in the ground truth model from the overall dataset. We used this dataset for training and validation purposes. The validation dataset was generated in a similar way containing 10 percent of all the mashups. The rest of the dataset was used for training purposes, which did not have API innovations of the ground truth model and the validation set. In the evaluation time, we observed whether the models recommended web APIs for the test mashups from their associated web API list in the ground truth model and measured their performance. For the precision and recall

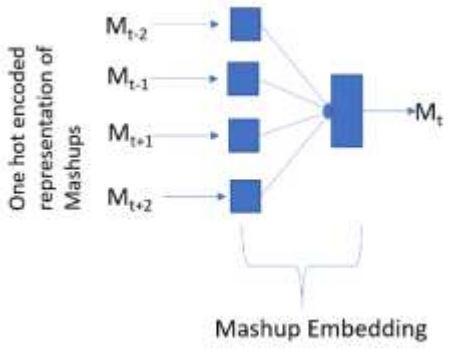

Figure 3. 1 Mashup Embedding Generation

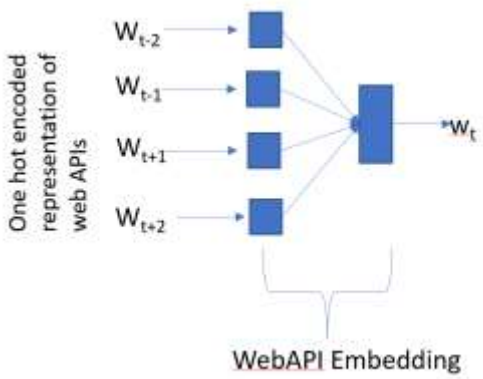

Figure 3. 2 WebAPI Embedding Generation

calculation, for each of the test mashups, we measured what percent of recommended APIs matched with the associated APIs from the ground truth model. For the NDCG value, for each of the test mashups, if the recommended API is in the associated API list from the ground truth model, then the relevance value of that API is one, otherwise zero. On the MF-based models, we tried with few random latent factor K sizes (100, 200, 500, 600). The results were close to each other, and the best result was found at K = 200. On the RME model, we used 1×10^{-1} for regularization term λ's value.

3.5.5 Experimental Setup for LSTM-Rec, CNN-Rec, NCF-Rec Models:

For these three neural network and deep learning based models, we followed the basic input and output structure. The input of the model is a mashup and API, and we used these representations to generate mashup and web API embeddings. Then we concatenated these embeddings together, and fed them to the neural network and deep learning based models. The output of these models are the probabilities of them being related and not related.

So, the basic layout is input (m, w) -> mashup and API Embeddings (Mt, Wt) -> Neural Network and Deep Learning based Models -> SoftMax Layer -> Output (p, q). Here, m is the one hot encoded representation of a mashup, a is the one hot encoded representation of an API. Mt is

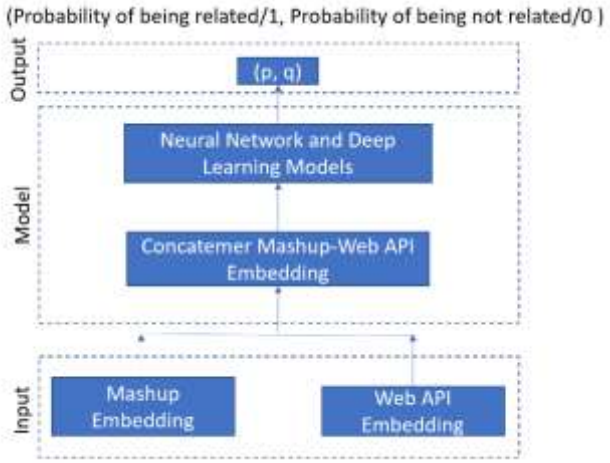

Figure 3. 3 Architecture of LSTM-Rec, CNN-Rec, NCF-Rec Models

the mashup embedding, and Wt is the web API embedding. Softmax layer is intended to normalize the model output to a probability distribution output. The output p is the probability of input m and a is being related, q is the probability of (m, a) is being not related. Figure 3.1, 3.2, and 3.3 gives an overall idea of the architecture for these models. We used same training, validation, and testing dataset for these models.

For, NCF-Rec model, we used the following architecture: Input Mashup and Web API Embeddings (Mt + Wt) ->LinearLayer (16, 21) -> TanhLayer(21, 21) -> LinearLayer (21, 8) -> TanhLayer(8,8)-> LinearLayer (8, 2) -> TanhLayer(2, 2)-> Softmax Layer () -> Output(p, q). Here, the first numbers represent the input dimension, and the second number represents the output dimension in each layer. The LinearLayer is a linear layer, and TanhLayer is the non-linear layer with Tanh activation function.

In this implementation of LSTM-Rec model, we used 50 units of the model with tanh activation function. In the recurrent unit, we used sigmoid activation function. Then we passed the output to a dense neural network which generated 2 outputs. The whole model is optimized on stochastic gradient descent, and MAE as loss function. For the CNN-Rec model, We used 1D convolutional

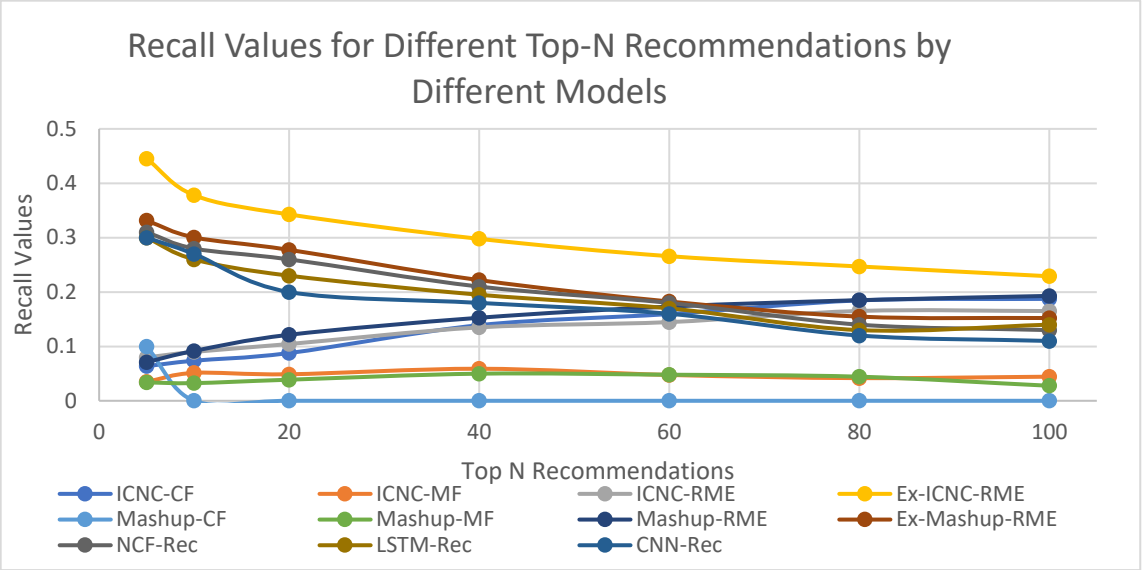

Figure 3. 4 Recall values at top N recommendations by the models.

model with 32 filters with kernel size is 2, and rectified linear unit as activation function. Both for the LSTM-Rec, and CNN-Rec a fully connected layer is added to predict the probabilities that they are connected or not.

3.5.6 Experimental Results

Figures 3.4, 3.5, and 3.6 summarizes the experimental results. We can see that our proposed model ExICNC-RME outperforms every other model in NDCG, precision, and recall value at all the top N recommendations. Our model achieved 0.47 NDCG value, where the second-best performing model Ex- Mashup-RME model achieved an NDCG value of 0.34, and all other models were below 0.1 for the top 5 web API recommendations. On average, our model achieved 0.35 NDCG value, where the second-best performing model ExMashupRME achieved 0.26 NDCG value, and all other models achieved below 0.13 NDCG value. In the case of precision and recall, the scenario is also similar. On average, our proposed model achieved 18.9% precision and 31.5% recall value, and the second-best model ExMashupRME achieved 12.3% precision and 23.2% recall value, all other models below 5.7% precision and 14.1% recall value. In terms of

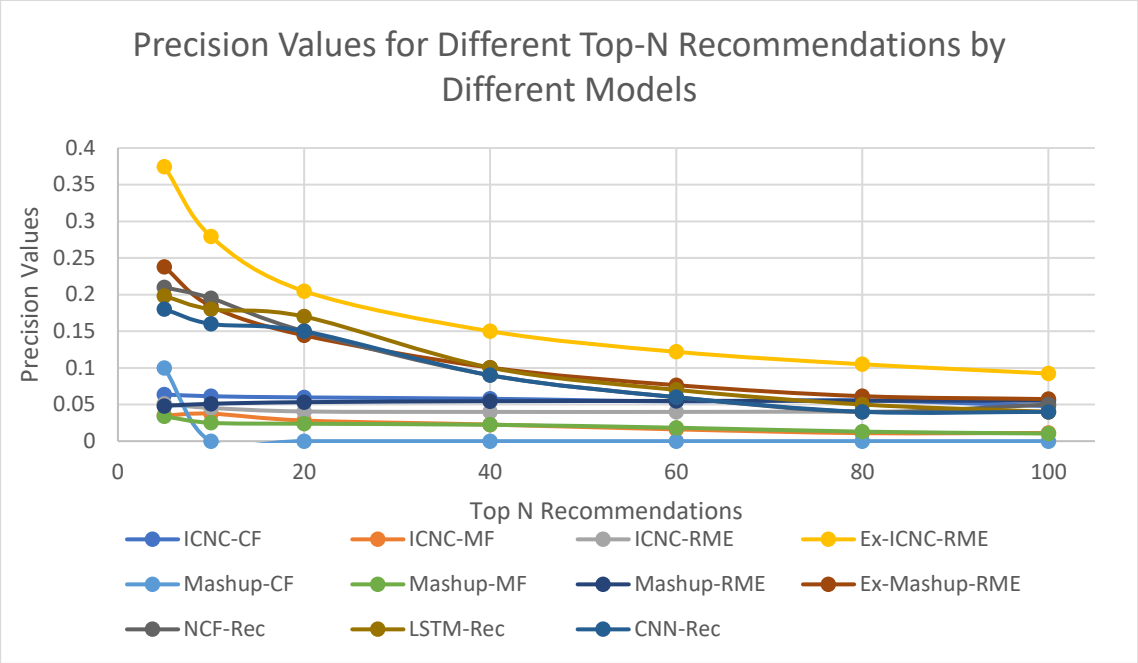

Figure 3. 5 Precision values for top N recommendations by the models.

improvement, our model achieved 54% more precise, achieved 36% more recall value, and 36.4% more NDCG value.

RME based models achieved better performance than the CF, MF, neural and deep learning based models. This shows the benefit of using mashup and web API specific additional information with the invocation data in the recommendation process as CF and MF based models only used invocation data. The comparison between the ICNC-RME and ExICNC-RME shows the importance of using user interaction data and the comparison between ExICNC-RME and ExMashup-RME shows the importance of using using two-level topic modeling data to generate a denser mashup-web API dataset. These results validate the effectiveness of using supplemental information and sparsity reduction techniques of our model in web API recommendation.

3.6 Discussions

User interaction with mashup and web API has been used to recommend APIs to the user [14] [15] and not to a mashup application like our model. If a user does not have any interaction history

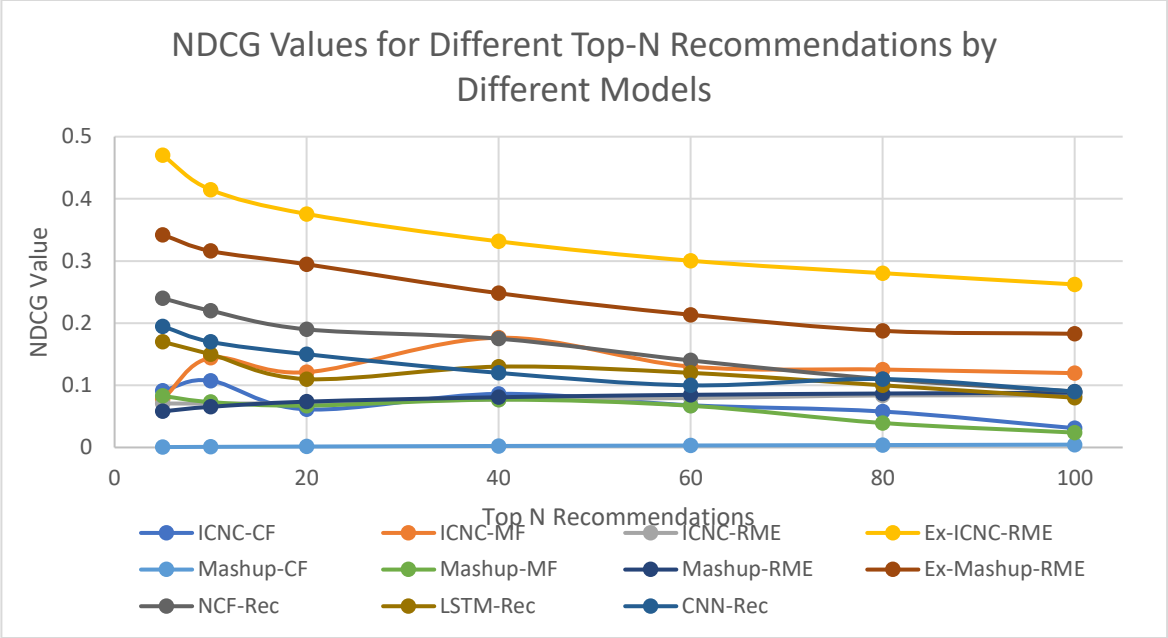

Figure 3. 6 NDCG values for top N recommendations by the models.

with the mashups or web APIs, these models would not be able to recommend APIs to these users. Also, user-based models filtered the recommendation results taking the user's explicit requirement [14] [15]; this kind of requirement data is not publicly available. Our developed model is not bounded by these user interaction histories or requirement data.

When mashup application developers want to build an application, there is no web API invoked. Our model can handle this cold-start problem to a certain extent. If there are similar mashups description wise, our model will use these web APIs from these mashups and add into the invoked APIs of this mashup. Also, if the invoked APIs have any related APIs in the global related APIs, our model will add these APIs into the invoked APIs of the new mashup. Then, our model will generate recommendations for this mashup. However, if the mashup application does not have any description or any similar mashups, our model will not be able to recommend APIs.

3.7 Conclusion

In this paper, we presented a novel web API recommendation model for the mashup application development process. Our model used two-level topic modeling and user interaction with web APIs and mashups to identify related web APIs and mashups. These two techniques are used sequentially to create a denser mashup-web API dataset. Then, we applied the RME model with the mashup and web API embeddings to generate effective web API recommendations. The experimental results show that our model achieves better accuracy and recommendation quality over other existing models. On average, our model achieved 54% more precision, 36.4% more NDCG, and 36% more recall value. We think our model outperformed these models by handling the low API invocation issue via creating a denser mashup - web API dataset and explicitly incorporating 'mashup to mashup' and 'web API to web API' information by the RME model. In the future, we plan to incorporate different other techniques to reduce sparsity, identify, and use more additional information on the web API recommendation.

3.8 References

[1] K.-B. Yue, "Experience on Mashup Development with End User Programming Environment," Journal of Information Systems Education, vol. 21, no. 1, Nov. 2019, [Online]. Available: https://aisel.aisnet.org/jise/vol21/iss1/10.

[2] "APIs show Faster Growth Rate in 2019 than Previous Years | ProgrammableWeb." [Online]. Available: https://www.programmableweb.com/news/apis-show-faster-growth-rate-2019-previous-years/research/2019/07/17. [Accessed: 04-Mar-2020].

[3] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "Collaborative Web Service QoS Prediction via Neighborhood Integrated Matrix Factorization," IEEE Transactions on Services Computing, vol. 6, no. 3, pp. 289–299, Jul. 2013, doi: 10.1109/TSC.2011.59.

[4] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "QoS-Aware Web Service Recommendation by Collaborative Filtering," IEEE Transactions on Services Computing, vol. 4, no. 2, pp. 140–152, Apr. 2011, doi: 10.1109/TSC.2010.52.

[5] X. Chen, Z. Zheng, Q. Yu, and M. R. Lyu, "Web Service Recommendation via Exploiting Location and QoS Information," IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 7, pp. 1913–1924, Jul. 2014, doi: 10.1109/TPDS.2013.308.

[6] L. Yao, Q. Z. Sheng, A. H. H. Ngu, J. Yu and A. Segev, "Unified Collaborative and Content-Based Web Service Recommendation," IEEE Transactions on Services.

[7] X. Liu and I. Fulia, "Incorporating User, Topic, and Service Related Latent Factors into Web Service Recommendation," 2015 IEEE International Conference on Web Services, New York, NY, 2015, pp. 185-192.

[8] W. Gao, L. Chen, J. Wu and H. Gao, "Manifold-Learning Based API Recommendation for Mashup Creation," 2015 IEEE International Conference on Web Services, New York, NY, 2015, pp. 432-439.

[9] B. Cao, X. F. Liu, M. M. Rahman, B. Li, J. Liu, and M. Tang, "Integrated Content and Network-Based Service Clustering and Web APIs Recommendation for Mashup Development," IEEE Transactions on Services Computing, vol. 13, no. 1, pp. 99–113, Jan. 2020, doi: 10.1109/TSC.2017.2686390.

[10] Y. Zhou, L. Liu, C. S. Perng, A. Sailer, I. Silva-Lepe and Z. Su, "Ranking Services by Service Network Structure and Service Attributes," 2013 IEEE 20th International Conference on Web Services, Santa Clara, CA, 2013, pp. 26-33.

[11] D. Soutas, D. Sacharidis, A. Simitsis and T. Sellis, "Ranking and Clustering Web Services Using Multicriteria Dominance Relationships," in IEEE Transactions on Services Computing, vol. 3, no. 3, pp. 163-177, July-Sept. 2010.

[12] J. Zhu, Y. Kang, Z. Zheng and M. R. Lyu, "A Clustering-Based QoS Prediction Approach for Web Service Recommendation," 2012 IEEE 15th International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops, Shenzhen, Guangdong, 2012, pp. 93-98.

[13] Y. Xu, J. Yin, Y. Li. "A collaborative framework of web service recommendation with clustering-extended matrix factorisation." in International Journal of Web and Grid Services, vol. 12, Issue 1, pp. 1-25, 2016.

[14] K. Fletcher, "Regularizing Matrix Factorization with Implicit User Preference Embeddings for Web API Recommendation," in 2019 IEEE International Conference on Services Computing (SCC), 2019, pp. 1–8, doi: 10.1109/SCC.2019.00014.

[15] L. Yao, X. Wang, Q. Z. Sheng, B. Benatallah, and C. Huang, "Mashup Recommendation by Regularizing Matrix Factorization with API Co-Invocations," IEEE Transactions on Services Computing, pp. 1–1, 2018, doi: 10.1109/TSC.2018.2803171.

[16] T. Tran, K. Lee, Y. Liao, and D. Lee, "Regularizing Matrix Factorization with User and Item Embeddings for Recommendation," Proceedings of the 27th ACM International Conference on Information and Knowledge Management - CIKM '18, Aug. 2018, doi: 10.1145/3269206.3271730.

[17] C. Zhao, C. Ma, J. Zhang, J. Zhang, L. Yi and X. Mao, "HyperService: Linking and Exploring Services on the Web," 2010 IEEE International Conference on Web Services, Miami, FL, 2010, pp. 17-24.

[18] Q. Wu, A. Iyengar, R. Subramanian, I. Rouvellou, I. Silva-Lepe, and T. A. Mikalsen. "Combining quality of service and social information for ranking services." Service-Oriented Computing. Springer Berlin Heidelberg, 2009, pp. 561–575.

[19] Y. Zhou, L. Liu, C. S. Perng, A. Sailer, I. Silva-Lepe and Z. Su, "Ranking Services by Service Network Structure and Service Attributes," 2013 IEEE 20th International Conference on Web Services, Santa Clara, CA, 2013, pp. 26-33.

[20] D. Skoutas, D. Sacharidis, A. Simitsis and T. Sellis, "Ranking and Clustering Web Services Using Multicriteria Dominance Relationships," in IEEE Transactions on Services Computing, vol. 3, no. 3, pp. 163-177, July-Sept. 2010.

[21] Y. Xu, J. Yin, Y. Li. "A collaborative framework of web service recommendation with clustering-extended matrix factorisation." in International Journal of Web and Grid Services, vol. 12, Issue 1, pp. 1-25, 2016.

[22] Q. Yu, Z. Zheng, and H. Wang, "Trace norm regularized matrix factorization for service recommendation," in 2013 IEEE 20th International Conference on Web Services (ICWS), 2013, pp. 34–41.

[23] X. Chen, Z. Zheng, X. Liu, Z. Huang, and H. Sun, "Personalized qos-aware web service recommendation and visualization," Services Computing, IEEE Transactions on, vol. 6, no. 1, pp. 35–47, 2013.

[24] M. Jamali and M. Ester, "A matrix factorization technique with trust propagation for recommendation in social networks," in Proceedings of the fourth ACM conference on Recommender systems, 2010, pp. 135–142.

[25] N. N. Liu, B. Cao, M. Zhao, and Q. Yang, "Adapting neighborhood and matrix factorization models for context aware recommendation," in Proceedings of the Workshop on Context-Aware Movie Recommendation, 2010, pp. 7–13.

[26] M. M. Rahman, X. Liu, and B. Cao, "Web API Recommendation for Mashup Development Using Matrix Factorization on Integrated Content and Network-Based Service Clustering," in 2017 IEEE International Conference on Services Computing (SCC), 2017, pp. 225–232, doi: 10.1109/SCC.2017.36.

[27] Q. Xue, L. Liu, W. Chen, and M. C. Chuah, “Automatic generation and recommendation for api mashups,” in Machine Learning and Applications (ICMLA), 2017 16th IEEE International Conference on. IEEE, 2017, pp. 119–124.

[28] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. Chua, “Neural collaborative filtering,” in Proc. Int. Conf. World Wide Web, 2017, pp. 173–182.

Chapter 4: Background, and Empirical Studies through Cyber Argumentation Platform

Dr. Xiaoqing (Frank) Liu and his previous research team worked on the cyber-argumentation system and different core components of such a system over a couple of years [1, 2, 3, 4, 5]. These core components include structured discussion, argument reduction techniques [1, 2]. With some enhancements, we developed an Intelligent Cyber Argumentation System (ICAS), which we later used to conduct two empirical studies and collect different datasets for our research. The first empirical study was performed at the spring 2018 session. I did not contribute to the system updates for the spring 2018 empirical study. For the spring 2019 empirical study, some new features were added in the system, such as social networking features, deliberative polling among participants, notifications to users, etc. for the spring 2019 empirical study. I contributed to this iteration of updates for the system, along with three other graduate students in our research group. Dr. Douglas Adams and Joseph Sirrianni made a significant contribution to conduct the empirical studies, developing different features, and maintain the system throughout the empirical studies.

4.1 ICAS System

In ICAS, discussions take on a tree structure architecture wise. At the root of each discussion, there is a core issue, which describes the overarching discussion problem to address. Under the issue, there are several different positions for discussion in our system. Each position is a different

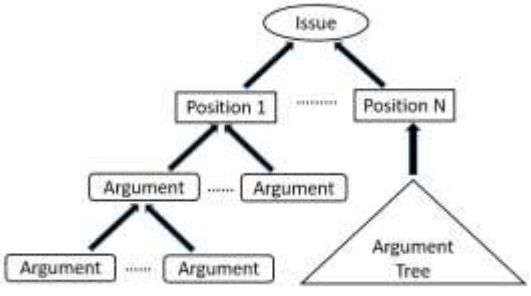

Figure 4. 1 The tree structure for discussions in ICAS.

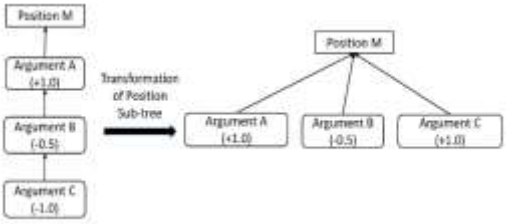

Figure 4. 2 Example of an argument reduction. Argument B, C are reduced from the second level of the tree to the first level.

perspective/solution which addresses or provides solutions to the parent issue. All discussions take place under a position where users can make arguments, support, or attack the parent position or other users' arguments. In the tree structure, issues are the root nodes of the tree, the issue's positions are first-level nodes of the tree, and all the arguments made by different users in a position

Table 4. 1: Empirical Dataset Description

Issue Name	Positions
Guns on Campus: "Should students with a concealed carry permit be allowed to carry guns on campus?"	No, college campuses should not allow students to carry firearms under any circumstances.
	No, but those who receive special permission from the university should be allowed to concealed carry.
	Yes, but students should have to undergo additional training.
	Yes, and there should be no additional test. A concealed carry permit is enough to carry on campus.
Religion and Medicine: "Should parents who believe in healing through prayer be allowed to forgo medical treatment for their child?"	Yes, religious freedom should be respected.
	Yes, but only in cases where the child's life is not in immediate danger.
	No, but may deny preventative treatments like vaccines.
	No, the child's medical safety should come first.
Same Sex Couples and Adoption: "Should same sex married couples be allowed to adopt children?"	No, same sex couples should not be allowed to legally adopt children.
	No, but adoption should be allowed for blood relatives of the couple, such as nieces/nephews.
	Yes, but same sex couples should have special vetting to ensure that they can provide as much as a heterosexual couple.
	Yes, same sex couples should be treated the same as heterosexual couples and be allowed to adopt via the standard process.
Government and Healthcare: "Should individuals be required by the government to have health insurance?"	No, the government should not require health insurance.
	No, but the government should provide help paying for health insurance.
	Yes, the government should require health insurance and help pay for it, but uninsured individuals will have to pay a fine.
	Yes, the government should require health insurance and guarantee health coverage for everyone.

are the rest nodes of the tree. Figure 4.1 gives a visualization of this tree structure. Participants can engage in discussion by making arguments directly to the position or another user's argument. When a user makes an argument, they fill out two fields. First is the text of the argument, which contains the rationale of the users for their support/attack to the parent node. Second is the level

of agreement with the parent node. An argument's level of the agreement indicates how much a user agrees or disagrees with the parent argument or position. Users can choose their level of agreement value from a weighted scale ranging from -1.0 to +1.0 at 0.2 intervals. The sign of agreement value specifies whether the user agrees (+ ve), disagrees (- ve), or is indifferent (0) toward the parent node. And the value specifies the intensity of the agreement or how much a user agrees or disagrees with it.

4.2 Deriving User's Opinion using ICAS

The user's opinion value on a position is calculated using the agreement values from all the posted arguments by that user under that corresponding position. But not all the arguments are made directly to the position; an argument can be made to another user's argument in the argumentation tree. So, we first need to connect the arguments that are further down the argument tree (past the second level) to the root position since their agreement values relate to other arguments, instead of the position itself. ICAS's built-in argument reduction method [1, 2] handles this process. The argument reduction method reduces an argument from any level of the argument tree to the first level. It calculates the user's agreement value directly towards the root position using artificial intelligence, fuzzy logic, linguistic heuristic rules, and other techniques. Figure 4.2 visualizes this reduction. For a more in-depth explanation of the fuzzy logic engine and argument reduction method, refer to [1, 2, 3, 4, 5, 6].

4.3 Empirical Study and Dataset

We organized empirical stud in an entry-level sociology class in the spring of 2018 and 2019 session. Students were asked to participate in this empirical study to discuss different social issues for five weeks' time span. The study contained four issues, and each issue had four different positions. The students were asked to contribute at least ten arguments in each issue. Table 4.1 contains the issues and positions in the empirical dataset.

4.4 User Participation and Dataset Statistics

4.4.1 User – Argument Dataset Statistics

The following table contains the number of users who contributed to the discussion and the number of arguments in the spring 2018 and spring 2019 sessions.

Table 4. 2 User and Argument Statistics

	Spring 2018	Spring 2019
Number of Arguments	10609	6428
Number of Users	308	251

4.4.2 Reply and Reaction Dataset Statistics

The following table contains the reply and reaction statistics in the spring 2018 and spring 2019 empirical study. In summary, we have 7237 parent arguments which the participating users either replied to or reacted from the spring 2018 and spring 2019 empirical dataset.

Table 4. 3: Reply and Reaction Dataset Statistics

		Spring 2018 & 2019
Reply	Number of Parent Arguments	5029
	Number of Users	361
React	Number of Parent Arguments	2208
	Number of Users	267

4.5 Reference

[1] X. Liu, E. Khudkhudia, L. Wen and V. Sajja, An Intelligent Computational Argumentation System for Supporting Collaborative Software Development Decision Making, 2010, pp. 167–180. doi:10.4018/978-1-60566-758-4.ch009.

[2] S. Sigman and X.F. Liu, A computational argumentation methodology for capturing and analyzing design rationale arising from multiple perspectives, Information & Software Technology 45 (2003), 113–122. doi:10.1016/S0950-5849(02)00187-8.

[3] X.F. Liu, S. Raorane and M.C. Leu, A Web-based Intelligent Collaborative System for Engineering Design, in: Collaborative Product Design and Manufacturing Methodologies and Applications, W.D. Li, C. McMahon, S.K. Ong and A.Y.C. Nee, eds, Springer Series in Advanced Manufacturing, Springer London, London, 2007, pp. 37–58. ISBN 978-1-84628-802-9. doi:10.1007/978-1-84628-802-9_2.

[4] X.F. Liu, E.C. Barnes and J.E. Savolainen, Conflict Detection and Resolution for Product Line Design in a Collaborative Decision Making Environment, in: Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work, CSCW '12, ACM, New York, NY, USA, 2012, pp. 1327–1336, event-place: Seattle, Washington, USA. ISBN 978-1-4503-1086-4. doi:10.1145/2145204.2145402.

[5] R.S. Arvapally and X.F. Liu, Polarisation assessment in an intelligent argumentation system using fuzzy clustering algorithm for collaborative decision support, Argument & Computation 4(3) (2013), 181–208. doi:10.1080/19462166.2013.794163. https://content.iospress.com/articles/argument-and-computation/794163.

Chapter 5: Cross-issue Correlation based Opinion Prediction in Cyber Argumentation

5.1 Abstract

One of the challenging problems in large scale cyber-argumentation platforms is that users often engage and focus only on a few issues and leave other issues under-discussed and under-acknowledged. This kind of non-uniform participation obstructs the argumentation analysis models to retrieve collective intelligence from the underlying discussion. To resolve this problem, we developed an innovative opinion prediction model for a multi-issue cyber-argumentation environment. Our model predicts a user's opinions on the non-participated issues from similar users' opinions on related issues using intelligent argumentation techniques and a collaborative filtering method. Based on our detailed experimental results on an empirical dataset collected using our cyber-argumentation platform, our model is 21.7% more accurate, handles data sparsity better than other popular opinion prediction methods. Our model can also predict opinions on multiple issues simultaneously with reasonable accuracy. Contrary to existing opinion prediction models, which only predict whether a user aggress on an issue, our model predicts how much a user agrees on an issue. To our knowledge, this is the first research to attempt multi-issue opinion prediction with partial agreement in the cyber-argumentation platform. With additional data on non-participated issues, our opinion prediction model can help the collective intelligence analysis models to analyze social phenomena more effectively and accurately in the cyber argumentation platform.

5.2 Introduction

In modern times, people discuss different social and political issues interacting with each other on many online platforms. Although most of the online discussions take place on social media

platforms, these platforms are not designed for large scale discussions. An effective discussion platform should promote a healthy exchange of ideas and opinions among the participants and facilitate participants to be well informed. However, due to the unstructured platform design, social media platforms do not facilitate effective discussions among users. On the other hand, Cyber-Argumentation platforms are specially designed for effective large scale discussions among participants. In these platforms, participants come together and express their opinions, criticize and respond to each other's opinions, ideas, etc. in an organized structure, which helps to achieve a well-informed and effective discussion among the participants.

In order to facilitate large scale discussions, cyber-argumentation platforms impose explicit discussion structure with different argumentation frameworks. Some of the well-known argumentation frameworks are Dung abstract frameworks [1], IBIS [2] and Toulmin's model of argumentation [3] etc. These argumentation structures allow users to navigate and follow the discussion easily. These structures also help the argumentation analysis tools to effectively analyze the discussion. Argumentation analysis tools can capture the collective intelligence and reveal different hidden insights from the underlying discussion. These tools analyzed different social phenomena successfully in this environment. As example: identifying group-think [4], polarization [5], assessing argument validity [1], credibility [6] etc.

However, to effectively work, argumentation analysis tools require the issue discussions have through participation, and users express their opinions on all the issues explicitly, which can be comprehended by these analysis tools. However, this is not a usual scenario; not all the issues get uniform participation from the users. Typically users participate only on few issues and do not engage in discussion in other issues in the system. Existing opinion analysis models focus mostly on analyzing user opinion on the participated issues only [7] [8]; often, the scope of such analysis

is limited. These missing opinion values on the non-participated issues may be crucial, and discarding these values may yield an incomplete analysis of the underlying discussion.

Few research attempts have been made in predicting user opinion on non-participated issues in the cyber argumentation environment [9] [10]. The accuracy of these opinion prediction methods is a significant concern, as predicted opinion values with lower accuracy will introduce error and misinformation in the analytical models. Another major concern is that these research works only attempted to predict whether a user would agree or disagree with an issue, not how much a user would agree/ disagree with the issue. Precise and detailed opinion values with varying levels of agreement/ disagreement are often required in many argumentation phenomena and opinion analysis models. An analysis of how much people's opinions might be influenced, not just whether people's opinions would be influenced; how controversial an issue might be, not just whether an issue might be controversial, etc. are some of the examples. Binary opinion prediction with a "yes/no" value can not fulfill the requirements for such analysis. To our best knowledge, no research attempt has been made, which predicts how much a user might agree on a non-participated issue in a cyber-argumentation environment.

We can solve this problem by predicting users' opinions with partial agreement on the issues that they have not explicitly expressed their opinion in discussion with high accuracy. We can generate a complete and detailed user-opinion dataset with a reasonably accurate prediction of missing information. Individual and collective opinion analysis of users can be conducted more precisely and effectively with more detailed opinion information, even if the user did not participate in some of the discussion. Detailed opinion prediction can help the complex argumentation analysis models such as group influence level in opinion dynamics, ingroup-outgroup activity [11], etc. Also, it can

help the collective intelligence assessment process more accurately, even when the discussions are incomplete.

In this paper, we present an opinion prediction method for a multi-issue cyber argumentation platform that predicts user opinion with partial agreement values on different ideas that they have not explicitly expressed their opinions. We used our argumentation platform, the Intelligent Cyber Argumentation System (ICAS), in which user participated in different issue discussions. We collected user opinions on issues from the discussions and predicted the missing opinions in non-participated issues. In our system, discussions take on a tree structure. Issues are the root of the conversation. Under an issue, there is a finite set of different positions that address the issue. For example, in the issue "Should guns be allowed on college campuses?" a position would be "Yes, because they would keep students safe." The participants then argue for or against the various proposed positions with complete or partial agreement/disagreement. We retrieved user opinion from the position discussion and predicted user opinion using our opinion prediction method in the non-participated position of different issues.

We developed a Cosine Similarity with position Correlation Collaborative Filter (CSCCF) model for opinion prediction with partial agreement. CSCCF is a collaborative filtering (CF) based model that predicts how much a user might agree with a particular position under an issue exploiting opinion correlation in different positions. We compared our CSCCF model with other opinion prediction methods based on popular predictive techniques on an empirical dataset, which we collected using our argumentation platform, ICAS. This dataset has four issues and sixteen associated positions, and it contains over ten thousand arguments in these discussions from more than three hundred participants. Different experimental results show that our model achieved good accuracy and 21.7% more accurate on average than other benchmarking predictive methods for

opinion prediction. With detailed experimental analysis, we evaluated the novelty of our CSCCF model over other comparison models in predicting opinion and analyzed different factors that impact the CSCCF model's prediction accuracy. In this work, we also analyzed group-representation phenomena in an issue discussion with predicted opinion values generated by the CSCCF opinion prediction model. We make the following contributions in this paper:

We make the following contributions in this paper:

- We proposed CSCCF model for cyber-argumentation, which uses user opinion similarity based collaborative filtering and opinion correlation between positions to predict user opinions on non-participated positions.
- We compared the accuracy of the CSCCF model with other popular opinion prediction techniques and different collaborative filtering based methods on an empirical dataset. Experimental results show that the CSCCF model is more accurate in different levels of sparsity in the dataset. CSCCF model can also leverage the correlation values present in the dataset in a better way than other comparison models in opinion prediction.
- With different experiments, we analyzed the impact of different key factors such as correlation degree, training data size, prediction multiple positions on the accuracy of the CSCCF model.
- We analyzed group-representation phenomena in the discussion to showcase how the CSCCF opinion prediction model can be used in our cyber argumentation platform.

The rest of the paper is organized as follows. We discussed about our argumentation system and how we mine user opinion to give a background for our work. Then we described the CSCCF

opinion prediction model, experimental data, results, and analysis. After this, we described the group-representation phenomena analysis and concluded the work.

5.3 Background

In this section, we discussed about our cyber-argumentation platform and how our platform derives user opinion from the underlying discussion data. This brief discussion will provide background information for our opinion prediction model presented in this paper.

5.3.1 ICAS System

In our previous research, we developed an Intelligent Cyber Argumentation System (ICAS), where participants can join and engage with each other to discuss different issues and associated positions [12]. The system can derive collective opinion on the position based on his/her arguments in the discussion. With some enhancements, we used this system to collect the user-opinion dataset for our research. Details of the system argumentation architecture can be found at section 4.1.

5.3.2 Deriving Viewpoint Vectors using ICAS

We represent a user's opinions in different positions using a viewpoint vector; this is a vector of numerical values where each element represents a user's opinion toward the position being discussed. User's opinion value on a position is calculated using the agreement values from all the posted arguments by that user under that corresponding position. ICAS's built in argument reduction method [12] [13] handles this process. For a more in-depth explanation of the fuzzy logic engine and argument reduction method, refer to [12, 13, 14, 15, 16].

Once all of the arguments have been reduced to the first level of the position sub-tree, then a user's opinion toward the position can be calculated by averaging the user's reduced agreement values to the position from all of their posted arguments. This process gives the user's opinion value

toward that position, which is added to the user's viewpoint vector at the corresponding index. If a user does not have any arguments under a position, then their entry at that index is missing, and we want to predict this value.

5.4 Opinion Prediction Model

In this section, we described our collaborative filtering based opinion prediction model with a brief discussion on data requirement and time complexity analysis of our model. Collaborative filtering based models identify the most similar users/items and predict the rating value from similar ones' rating patterns. In our case, items are different positions on the issues, and rating value is the user opinion value in the positions. We will find out the most similar user of the target user to predict what would be the target user's opinion on a position of an issue.

5.4.1 Data Required for Prediction

We use a viewpoint vector to represent the user opinion in different positions across issues. If a user did not participate in a position discussion, the associated opinion value in the viewpoint vector would be missing, and we want to predict this opinion value. We will use our opinion prediction model CSCCF to predict this missing user opinion. If a user x did not participate in position t, we need the following information to predict user x's opinion value at position t:

1) Viewpoint vectors of each user in training data, 2) Viewpoint correlations of opinion values between target position t with all other positions, and 3) Viewpoint vector of the target user (User x).

A user's viewpoint vector represents the user's opinion across all the positions of issues. Our model calculates viewpoint vectors for every user in training data. If there are n position in the system, the viewpoint vector of user i can be represented in the following format:

$U_i = [R_1^i, R_2^i, R_3^i, R_4^i \ldots\ldots R_n^i]$; Here, U_i is the viewpoint vector of user i. R_p^i is user i's opinion at position p. If the user i did not participate in position p, then R_p^i will be represented as an invalid or missing value.

The viewpoint correlation of opinion values between a target position and all other positions is a vector of correlation values. Each of the values represents the correlation degree to which the opinion values in the target position are relater with another position. In our system, opinions across all positions have the same value range from -1.0 to +1.0. So, a strong positive correlation between two positions indicates that overall users have similar opinions in these two positions, users agree or disagree with similar agreement level in these two positions. Likewise, a strong negative correlation indicates that users have opposing opinions in these two positions; if users agree in one position, they will disagree in another position with a nearly similar intensity. Whereas a weak correlation value between two positions does not indicate any linear relationship between users' opinions in these two positions. Correlations between position i and j is calculated using users opinion at position i and j with Pearson Correlation Coefficient using equation 1. We only included correlation values with high confidence values, correlation values with lower confidence values determined from two-tailed p-values above or equal to 0.05 are set zero.

$$C_{i,j} = \begin{cases} \frac{\sum_{k=1}^{m}(R_i^k - \hat{R_i})*(R_j^k - \hat{R_j})}{\sqrt{\sum_{k=1}^{m}(R_i^k - \hat{R_i})^2} * \sqrt{\sum_{k=1}^{m}(R_j^k - \hat{R_j})^2}} & if\ p < 0.05 \\ 0 & else \end{cases} \qquad (1$$

Correlation values for a position t with other positions can be represented in the following way:

$C_t = [C_{t1}, C_{t2}, C_{t3}, \ldots, C_{tn}]$; Here, C_t is the correlation value vector of position t. C_{ti} is the pearson correlation coefficient value between position t and position i.

We can represent the target user (user x)'s viewpoint vector in the following vector format:

$U_x = [R_1^x, R_2^x, R_3^x, R_4^x \ldots, R_{t-1}^x, ?, R_{t+1}^x \ldots\ldots R_n^x]$; Here, user x's opinion value at position t or R_t^x is missing, we will predict this value in the following section.

5.4.2 CSCCF Opinion Prediction Model

We want to predict user x's opinion value at position t or the value of R_t^x in the viewpoint vector of Ux. We predict this value by integrating the opinion values of most similar users with respect to position t. There are two steps in this process. First, we will measure the opinion similarity of user x with every other user who does not have a missing value at position t in the training data. Second, we will integrate the topmost similar users' opinion values at position t as the predicted value for user x at position t; R_t^x.

To measure the similarity between our target user x and other users in the training dataset, first we will remove any user who has a missing value at position t. The rest of the users who do not have a missing value at position t, are placed into user x's candidate set. Then we measure the similarity between user x and every user in the user x's candidate set.

We will calculate the similarity between user x and user y. The viewpoint vectors of user x and user y are U_x and U_y.

$U_x = [R_1^x, R_2^x \ldots R_{t-1}^x, ?, R_{t+1}^x \ldots\ldots R_n^x]$

$U_y = [R_1^y, R_2^y \ldots R_{t-1}^y, R_t^x, R_{t+1}^y \ldots\ldots R_n^y]$

First, we will remove the elements from the viewpoint vectors in which either one has a missing value. In this case, U_x has a missing value at position t, so we will remove $R_t x$ and $R_t y$ from the viewpoint vectors U_x and U_y.

$U_x = [R_1^x, R_2^x \ldots R_{t-1}^x, R_{t+1}^x \ldots\ldots R_n^x]$

$U_y = [R_1^y, R_2^y \ldots R_{t-1}^y, R_{t+1}^y \ldots\ldots R_n^y]$

In the next step, we use the correlation values from the target position t's correlation vector Ct to update the viewpoint vectors. Each value in the viewpoint vector is multiplied with it's corresponding correlation value with the target position t. For example, the opinion values at position i R_i^x and R_i^y will be multiplied by the opinion correlation value between position i and position t, C_{ti}. The updated viewpoint vectors are represented using the following notations $U_x^\wedge$ and $U_y^\wedge$.

$U_x^\wedge = [C_{t,1}R_1^x, C_{t,2}R_2^x, .. , C_{t,t-1}R_{t-1}^x, C_{t,t+1}R_{t+1}^x, ..,C_{t.n}R_n^x]$

$U_y^\wedge = [C_{t,1}R_1^y, C_{t,2}R_2^y, .. , C_{t,t-1}R_{t-1}^y, C_{t,t+1}R_{t+1}^y, ..,C_{t.n}R_n^y]$; Here, the opinion value at position i is multiplied by the correlation value with position t, C_{ti}.

Next, we calculate the cosine similarity between the updated viewpoint vectors $U_x^\wedge$ and $U_y^\wedge$ using equation 2. The cosine similarity value is used to determine how similar user x and user y are with respect to position t. The range of cosine similarity value is in between -1 to +1. Here, +1 represents the two vectors are completely similar to each other, 0 represents that the vectors have no correlation, and -1 represents that two vectors are completely different to each other.

$$Similarity(x,y) = Cosine\ Sim\ (U_x^\wedge, U_y^\wedge) = \frac{\sum_{i=1, i\neq t}^{n} C_{ti}^2 R_i^x R_i^y}{\sqrt{\sum_{i=1,i\neq t}^{n} C_{ti}^2 (R_i^X)^2} + \sqrt{\sum_{i=1,i\neq t}^{n} C_{ti}^2 (R_i^y)^2}} \qquad (2)$$

In this way, we measure the similarity between user x and every other user in x's candidate set. Once we measure the similarity with all users, we sort and rank the users based on their similarity value with our target user x. Then we select the top k most similar users, here k is a constant model parameter. We investigated with different values for the value of k such as 3, 5, 10 etc. and we got the most accurate result when the value of k was set at 5 on our validation dataset. Our model then

integrates the opinion value at position t from the top k most similar neighbors; it averages the R_t opinion value weighted by the similarity value using the following equation, as shown in (2).

$$\textit{Predicted value of } R_t^x = \frac{\sum_{m=1}^{k} Similarity(x,m) * R_t^m}{\sum_{m=1}^{k} Similarity(x,m)} \quad (3)$$

Our model measures the similarity between two users based on which position we are predicting. We multiply the opinion values with their correlation value with the test position. It weights or prioritizes the opinion values based on how important they are in determining the test position. If we predict another position s, the topmost similar users for target user x will be different than the topmost similar users when we predict position t. Also, our model filters out uncorrelated opinion values by multiplying them by zero or near zero in the similarity calculation.

5.4.3 Time Complexity of our CSCCF model

In this time complexity analysis, we will measure the time complexity of our model to predict a single opinion value for a test user. The time complexity of calculating the correlation values from the training data is not included in this measurement as we perform this step only one time in the beginning and use it to predict opinion values for all test users. We will use n as the number of users n and p as the number of positions. In the prediction process, we update the viewpoint vectors with the correlation values; then, we measure the cosine similarity value on the updated viewpoint vectors. The time complexity of this approach is O(n*p). In the next step, we sort the similarity values from n users and use the opinion values from top k users to make the prediction. The time complexity of this step is the time complexity of sorting n numbers. We used a heap-based priority queue, so the time complexity of our approach is O(nlogn). So, the overall time complexity of our algorithm to make one single prediction is O(n*p) + O(nlogn).

5.5 Experiments

5.5.1 Empirical Data Description

We organized an empirical study in an entry-level sociology class in the spring of 2018 session and used the collected dataset in this work.

5.5.2 Methods to test against

We implemented different popular opinion predictive techniques and compared their accuracy with our CSCCF model using the collected empirical dataset. These methods include one neural net, two matrix-factorization based approaches, and six different memory-based collaborative filtering models. The only difference between these CF models and our CSCCF model is how we calculate the similarity between two users. CSCCF and these CF models predict Opinion value from the most similar users in the same way. The following section briefly describes all the comparison models.

1) Cosine Similarity based CF (CSCF)

This method used the Cosine similarity between the original viewpoint vectors to select the topmost similar users. For two users x and y, their similarity is measured using their agreement vectors U_x and U_y with the following equation.

$$cosine\ similarity(U_x, U_y) = \frac{\sum_{i=1, i\neq t}^{n} R_i^x R_i^y}{\sqrt{\sum_{i=1, i\neq t}^{n} (R_i^X)^2} + \sqrt{\sum_{i=1, i\neq t}^{n} (R_i^y)^2}} \qquad (4)$$

As compared to our CSCCF model, this method does not consider correlation values with the target position; each value in the agreement vector has the same priority in the similarity calculation. In our CSCCF model, we measure similarity on the updated viewpoint vectors multiplied by correlation values with the target position. In the similarity calculation, more

correlated position values will have a higher value difference range than the less correlated ones, which indirectly incorporates the importance of more correlated indexes. Testing our method against this method highlights the importance of the position correlations when predicting different values.

2) Neural Net

Neural nets have been used extensively in research to solve complex problems, and have been modified to solve collaborative filtering problems too. We implemented a neural net model that uses hybrid latent variables as a hybrid collaborative filtering technique as described in [17] to learn individual information about both the users and items; in our case, items are positions. During the training phase, the neural net model learned the weights and latent input variables at the same time. We tried the neural net with various input layer vector sizes, the best result we got when the latent vectors were at length 2 for both users and positions. We used this topology in our neural net implementation: linear layer (4, 6) => Tanh layer (6, 6) => linear layer (6, 1) => Tanh layer (1, 1). Here, the first parameter is the input size, and the second parameter is the output size in different layers. The model attempted to predict the user's opinion, given a user's latent vector and a position's latent vector. The neural net used stochastic gradient descent to update the weight parameters, and sum squared error (SSE) was used to optimize the neural net.

3) Matrix Factorization

Matrix factorization is a popular predictive method that decomposes a matrix into multiple matrixes such that when they are multiplied together, it returns the original matrix. We implemented a Regularized Incremental Simultaneous MF as described in [18], which applies regularization techniques via penalizing the magnitude of vectors to avoid overfitting. In our case,

the original matrix is the user-position matrix (R = |U | * |D|) which is further broken down in two matrices (P = |U|*|K| and Q = |D| * |k|) to discover K latent features associated with users and positions.

$$R \approx P \times Q^T \qquad (5)$$

We tried different latent factor sizes for K, and the best result was found when K was 5. Our matrix factorization model was also optimized for the sum squared error.

4) Probabilistic Matrix Factorization

We implemented Probabilistic Matrix Factorization (PMF) as described in [19]. PMF is a Matrix Factorization based model that uses a probabilistic linear model and considers Gaussian observation noise. Like with the neural network and matrix factorization, PMF assumes users and positions have latent vectors of size k. However, unlike matrix factorization, the latent matrices are drawn from a normal distribution, determined by the means and variances of each row in the original matrix. So, when they are multiplied together, the resulting matrix is also normally distributed. The resulting matrix is derived in (5).

$$R \approx N(P \times Q^T, \sigma^2) \qquad (6)$$

Here P is the latent matrix for the user features, Q is the latent feature matrix for the positions, σ is the variance in the original training matrix. N is a function that samples from a Gaussian distribution defined by the product of P and Q^T with variance σ^2.

5) Spearman Rank Correlation Similarity based Collaborative Filtering (SRCSCF)

In this CF model, we used the original viewpoint vector (U_x and U_y) and sorted the opinion agreement values in different positions. Then, we used the indexes in the sorted viewpoint vector

as ranks of user's opinion values. Then with the ranks, we measured the similarity between user x and user y using the following equation:

$$Sim(u_x, u_y) = 1 - \frac{6 \sum_{h=0}^{n} d_h^2}{n(n^2-1)} \qquad (7)$$

Here, d_h is rank difference for an opinion at a position h between user x and user y. n is the number of positions at which both user x and user y participated or has valid opinion values.

6) Jaccard Similarity based Collaborative Filtering (JSCF)

This model measures the similarity between two users based on the number of items they rated with similar value. In our case, we rounded the opinion agreement values from the original viewpoint vector U_x and U_y up to two decimal points and checked whether the opinion values are similar or not. Then we measured the similarity between user x and user y using the following equation:

$$\text{Sim}(u_x, u_y) = J(U'_x, U'_y) = \frac{|U'_x \cap U'_y|}{|U'_x \cup U'_y|} \qquad (8)$$

7) Normalized Mean Squared Difference Similarity based Collaborative Filtering (NMSDSCF)

This model measures the Mean squared difference between the two original viewpoint vector U_x and U_y and then normalizes it with the maximum mean squared difference. Then it measures the similarity between two users using the following equation:

$$sim(u_x, u_y)^{NMSD} = 1 - Nomalized\ Mean\ Squared\ Difference(U_x, U_y) \quad (9)$$

8) Jaccard and Mean Squared Difference Similarity based Collaborative Filtering (JNMSDSCF)

This method uses the jaccard similarity and mean squared difference similarity and integrates them to measure the similarity between user x and user y using the following equation:

$$sim(u_x, u_y)^{JMSD} = sim(u_x, u_y)^{Jaccard} * sim(u_x, u_y)^{NMSD} \quad (10)$$

9) Pearson Correlation Similarity based Collaborative Filtering (PCSCF)

This model uses the Pearson correlation coefficient value calculated from the original viewpoint vector (U_x and U_y) to measure the similarity between user x and user y.

10) Constrained Pearson Correlation Similarity based Collaborative Filtering (CPCSCF)

This model is a modification of Pearson Correlation based Collaborative Filtering. It uses the midpoint of feature value instead of mean rating to measure the correlation and use it as the similarity between user x and user y.

5.5.3 Experimental Results

1) Predicting Opinion in a Single Position with Different Level of Sparsity

In this experiment, we analyzed the accuracy of our CSCCF and other baseline models in terms of Mean Absolute Error (MAE) when they predict user opinion in a single position. MAE value is calculated from the actual and predicted user opinion value at a particular position. We conducted this experiment in two variations of the dataset to evaluate accuracy in different level of sparsity. One variation of the dataset is the complete user-opinion dataset, where all users have opinion values in all the positions, and there are no missing values. Another variation is the entire user-opinion dataset, which is collected from the empirical study; this dataset contains missing values.

We performed a cross-validation test with five fold and two repetitions for each of the models and averaged the MAE values from the iterations. In each iteration, we divided the dataset as 80 percent training and 20 percent testing. Using this test environment, we evaluated the accuracy for each position and averaged the MAE values. This MAE value across all positions is reported in the experimental results. The following two sections contain the result from this experiment.

c) Accuracy on entire dataset

Figure 5.1 contains the accuracy values of different models in terms of MAE. From the results, we can see that CSCCF outperformed every other model in every position. The average MAE value over all the positions of CSCCF model is 0.133, whereas the MAE value from the second best-performing model PMF is 0.350. The MAE value from all other models lie in between 0.351 to 0.42. The MAE value for all other models was in between a narrow range. In contrast, the CSCCF model shows visible improvement filtering out uncorrelated opinion values and weighting related opinion values as per their importance to predict the test position. As an example, when we measured the MAE value for position 14, all comparison models hovered in between 0.31 to 0.39, but the CSCCF model achieved the MAE value of 0.09. From this experimental result, we can see that the CSCCF model outperformed all baseline comparison models, which show the importance of weighting the opinion values by their correlation values with the test position in the similarity calculation.

d) Accuracy on the dataset with no missing values

In this experiment, we compared the accuracy of CSCCF and other baseline models on the complete user-opinion dataset, where every user had opinion value in every position. Figure 5.2 contains a summary of this experiment. Compared to the MAE value on the entire dataset, the

MAE value of the CSCCF model decreased to 0.093 on this complete dataset. However, the MAE value of the second best performing model, which is PMF in this experiment got increased to 0.365. With few exceptions, the MAE of the comparison models tended to decrease in this complete dataset, especially for the CF-based models. So, less sparse data in the user feature vector is helping to find similar users more effectively. The MAE value of Matrix Factorization, Probabilistic Matrix Factorization, and Neural Net models increased than on the MAE value on the entire dataset. We think these models are suffering to figure out the latent relationship between users to their opinions because of the smaller data size in this dataset. Which is why the MAE value got increased compared to their MAE values on the entire dataset. The experimental result shows that the CSCCF model outperformed other models not only in the sparse dataset, it also outperformed these models in a complete dataset with no missing values.

e) Experimental Result Analysis:

The improvement over CF-based models, especially the Cosine Similarity based CF (CSCF), shows the usability of position correlations in similarity calculation. In CSCF, each position agreement value in the viewpoint vector has a similar priority when we measured the similarity between two users. Whereas in our CSCCF model, each opinion value is weighted according to

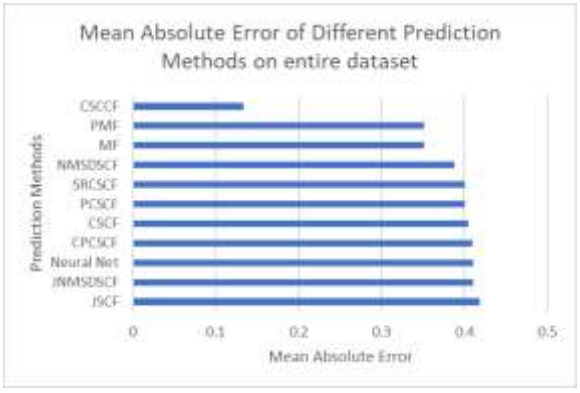

Figure 5. 2 Mean Absolute Error of different Models on entire dataset

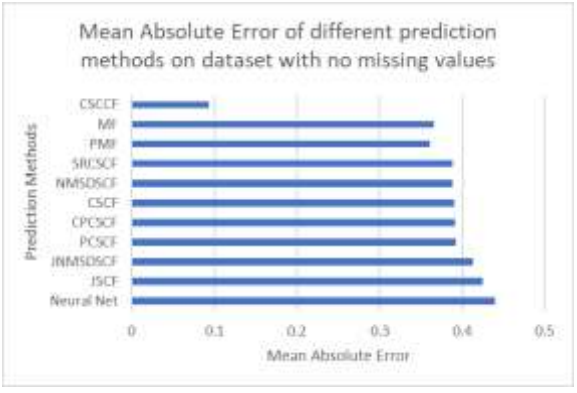

Figure 5. 1 Mean Absolute Error of different Models with no missing values

the correlation with the target position. Our model CSCCF also outperformed Neural Net, Matrix Factorization, and Probabilistic Matrix Factorization models. We think the main reason for this improvement is the limited data size. Our dataset contained lots of missing values as most of the users did not participate in all the position discussions. There is not a lot of values to learn about users and discover their pattern. So, the latent features and relationships learned by these models are most likely to be underdeveloped with little meaningful information resulting in lower accuracy.

Over model handles the data sparsity issue by utilizing the global correlation values calculated from training data and using them for each user with their limited available information. As there was not much data to learn about the individual user, our model made the best use of data by integrating the global correlation with users' personalized data points.

2) Predicting Opinion in multiple positions across Issues

In this experiment, we evaluated the accuracy of the CSCCF model when it predicts user opinion values at two to six positions simultaneously. With a five fold, two repetitions and 80:20 ratio for training and test data, We used all possible combinations while testing at each number of positions. As an example, when we predict two positions simultaneously, we experimented with all possible 120 two position indices combination, such as (0, 1), (0, 2) …. (14, 15) as testing positions and averaged the MAE values. We used this similar process to predict user opinion at 3 to 6 positions simultaneously. Figure 5.3 shows the result of this experiment on the complete dataset with no missing values. The MAE value increases with more positions being predicted at the same time. The average MAE value is relatively low up to 3 positions being predicted simultaneously; after three positions the MAE value increases at a faster rate. As our model uses the opinion values from the correlated positions to predict the opinion value at a position. If the positions being predicted

are correlated with each other, it will increase the MAE value of our model than when they are being predicted alone. Our model would not be able to use the opinion values from correlated positions to predict the test position as those positions are also being predicted simultaneously. This low data usage is affecting the MAE value of our model. As an example, if we predict position 1 and position 5 simultaneously, if position 1 and position 5 are correlated then the MAE value would be higher because when we predict position 1, our model won't be able to use the opinion value from position 5 and vice versa.

3) Predicting Opinion with Different Training Data Size

In our model, we calculate the correlation value between positions from the training data; the number of samples in the training data should have some impact on the overall accuracy of the CSCCF model. We evaluated the impact of varying training data sizes on the overall accuracy of the CSCCF model in this experiment. We divided the training and testing data in different ratios such as (80:20), (70:30), etc. and measured the MAE values at different training and testing data ratios. Figure 5.4 shows the MAE value of our model on the dataset with no missing values at different training data percentages. The smaller the training size, the larger the MAE value gets as some of the most similar users might be missing from training data. This rate increases after we include 70% of the users as training data but remains within 0.1 until we included 50% of the users

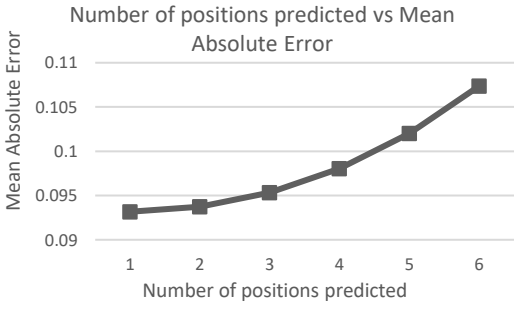

Figure 5. 4 Number of positions prediction vs Mean Absolute Error

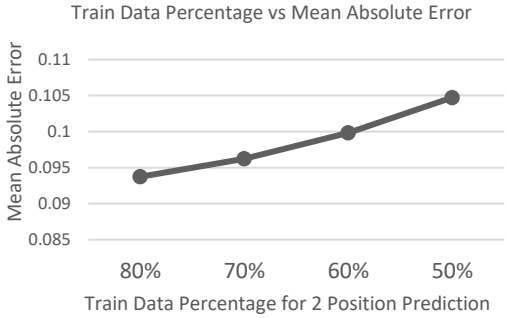

Figure 5. 3 Train Data Percentage vs Mean Absolute Error

in the training set. This test shows that even smaller sizes of training data do not affect the model drastically as a whole; it might affect individual positions.

4) Predicting Opinion by the Baseline Comparison Models on the Filtered dataset by Different Correlation Degree with different level of Sparsity

Our CSCCF model weighs the opinion values according to their correlation values with the test position in the similarity measurement between users. This step filters out uncorrelated opinion values and is the major contributing factor for the high accuracy achieved by the CSCCF model. In this experiment, we evaluated the impact of filtering the dataset by different correlation degree on the overall accuracy of the baseline models and whether filtering enables these baseline models to outperform CSCCF model.

To test this approach, we calculated the correlation values between the positions of different issues from the training data. Then we used the correlation values to filter out positions in the similarity calculation of collaborative filtering models. On a particular threshold correlation value, positions with greater or equal threshold correlation values were only used when calculating the similarity between users. For matrix factorization and probabilistic matrix factorization, agreement values in positions with lower correlation values with the test position were removed from the user-item matrix. This step will ensure that these values will not be used by these methods to predict the test position. The neural net model we implemented uses latent feature variables to learn about individual users and positions. In training time, for each (user, agreement value at a position) pair, it updates the associated latent user vector and latent position vector. At the end of the training, we have the final latent user vectors for each user and latent position vectors for each position. In testing time, for a (user, position) value pair, the associated user latent vector and position latent vectors are loaded to predict opinion. The idea of incorporating correlated data points in training

time is not valid here, as for a (user, position) pair (x, y), the x's latent vector and y's latent vector are used to update them. This step does not use all data points to enable us to incorporate y's correlated values only. For this reason, we did not include the neural net model in this experiment.

a) Accuracy on the entire dataset

We filtered the entire user-opinion dataset by different correlation values and measured the MAE values in this experiment. Figure 5.5 contains a summary of this experiment. For all CF models, the lowest MAE value was achieved by filtering the dataset with a threshold correlation value of 0.1; the MAE value at this point is significantly smaller than when the unfiltered dataset was passed to the CF-based models. After the 0.1 threshold correlation point, increasing correlation values resulted higher MAE values. The original dataset contains lots of noisy and irrelevant values. By filtering the dataset at the correlation value of 0.1, noisy values got removed from the dataset, which triggered the lowest MAE value for the collaborative filtering models. But further removing more data points by threshold correlation values is making the data size too small for collaborative filtering models to find users with reasonable similarity to derive or predict agreement value with high accuracy. For MF and PMF, the lowest MAE value was achieved by feeding the entire

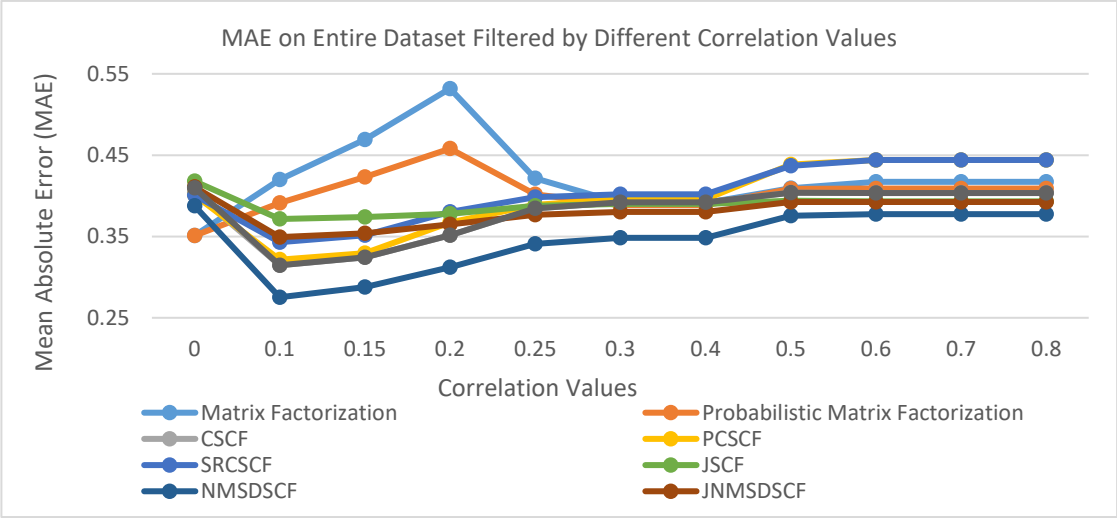

Figure 5. 5 MAE on entire dataset with different threshold correlation values

unfiltered dataset to the models. With each filter applied by threshold correlation value, the dataset got too small and probably lost meaningful information to figure out the latent features and relationships between users and items. This is why the MAE value was best when data was unfiltered than the filtered ones at different correlation values. None of these baseline models achieved high accuracy as the CSCCF model at all the threshold correlation degree. This experiment shows that even with filtering the dataset did not enable these baseline models to outperform CSCCF's accuracy on the entire dataset.

b) Accuracy on the Complete dataset with no Missing Values

In this experiment, we filtered the complete dataset with no missing value and feed them into different baseline models. Figure 5.6 contains a summary of this experiment. Matrix factorization and Probabilistic Matrix Factorization followed a similar pattern of the MAE values at different threshold correlation values. For both MF and PMF, the best MAE value was achieved by feeding the unfiltered dataset to the models. This complete dataset is already small in size; further filtering is making this dataset smaller in size. The smaller data size is affecting the learning process in MF and PMF, resulting in a lower accuracy in the filtered dataset than the unfiltered one. For all CF

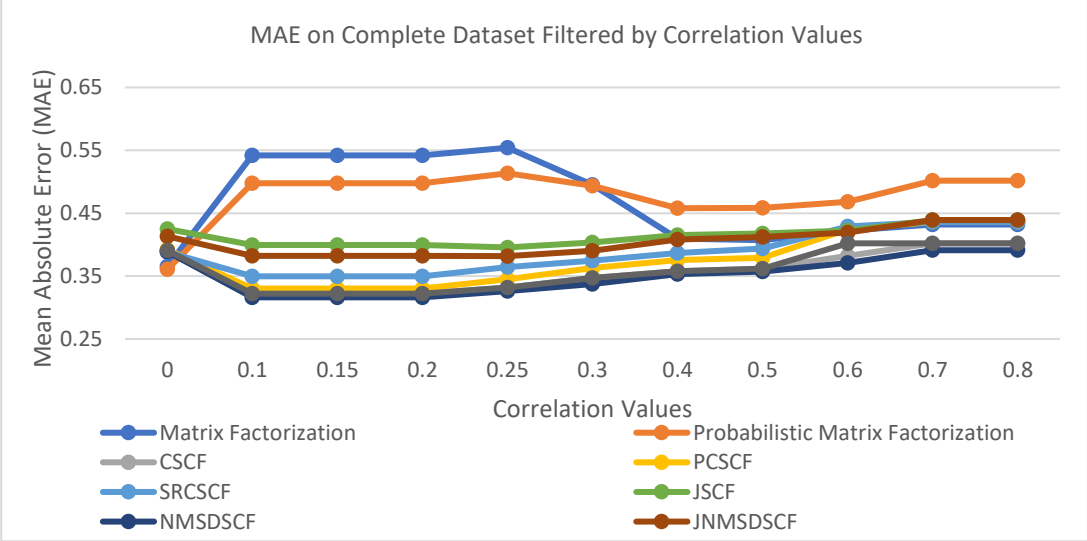

Figure 5. 6 MAE on complete dataset with different threshold correlation values

models except JSCCF, the best MAE value was achieved at the threshold correlation value of 0.1; after that, the MAE value increased gradually with increasing correlation values. None of the models did not outperform CSCCF's accuracy on the complete dataset in this experiment.

5) *Predicting Opinion by the Baseline Models on the Preprocessed dataset with Different Level of Sparsity*

In our CSCCF model, we multiplied the opinion values by their correlation values with the test position in the similarity measurement to prioritize data points according to their relevance with the test position. In this experiment, we analyzed the impact of feeding the weighed datapoints by correlation values to different baseline models and whether this step enables any of these models to outperform the CSCCF model. To analyze this scenario, we calculated the correlation values between different positions from the training data. Then for a particular test position, we multiplied the correlation values with the original agreement values in the training data. Then we measured the average MAE value on the modified dataset using the 80:20 training testing data ratio and five fold, two repetition cross-validation setup.

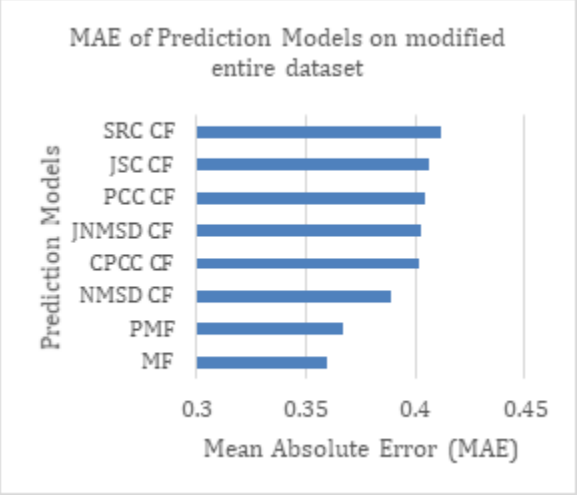

Figure 5. 7 MAE on prediction models on modified entire dataset

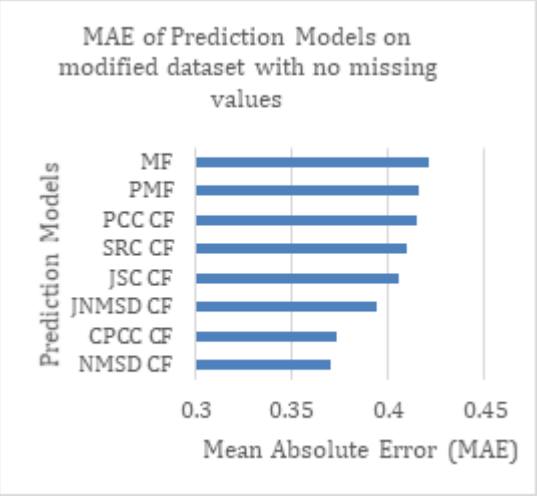

Figure 5. 8 MAE on prediction models on modified dataset with no missing values

Figure 5.7 and 5.8 contains the summary of this experimental result on the entire dataset and on the complete dataset respectively. On the entire dataset, MF and PMF achieved the lowest MAE value compared to other collaborative filtering models. However, on the complete dataset, MF and PMF achieved the worst MAE value out of all prediction models and NMSDCF achieved the lowest MAE value. The correlation-based CF models use correlation values as the similarity between users or items. The relationship cannot be extracted by further calculating correlation values on the modified by correlation data. This is the reason for the worse performance of these correlation-based CF models. On the complete dataset, even though the values were multiplied by the correlation values, the smaller size of the dataset is the reason we think MF and PMF did not achieve as low MAE value as on entire dataset. This also strengthens the fact that MF and PMF models needs more data to extract latent relationships between users to items to predict with high accuracy. None of these models outperformed CSCCF even with the modified dataset, which signifies the importance of weighing the data in similarity calculation as performed by the CSCCF model.

6) Determining Threshold Correlation Values for Reasonable Accuracy by CSCCF Model

Our CSCCF model relies on the correlation values on the dataset to predict opinion with reasonable accuracy. In this experiment, we tried to determine the threshold correlation value that needs to be present in the dataset for the CSCCF model to achieve high accuracy. At first, we measured the MAE value by the CSCCF model both on the entire dataset (with all 0's) and on the complete dataset (without any 0's). Then, we filtered both datasets by different correlation values and measured the corresponding MAE values to determine the threshold correlation value. Figure 5.9 summarizes the result of this experiment. From the results, we can see that the CSCCF model achieved the highest accuracy when both datasets were filtered by a correlation degree of 0.15.

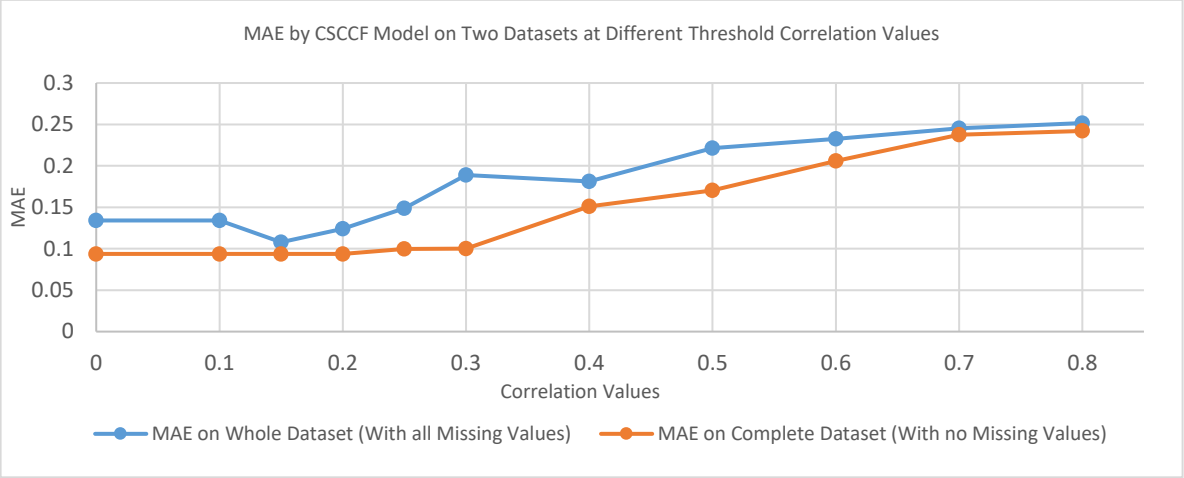

Figure 5. 9 MAE by CSCCF Model on Two Datasets at Different Threshold Correlation Values

Although, filtering by the higher correlation value should yield to lower MAE value, but it also lowers the percentage of data being used in the prediction model. This is the reason filtering by higher correlation is resulting in higher MAE values. A balance between filtering by a correlation degree and the percentage of data being used by the model needs to be considered. In our case, we utilized 80 percent or above of available data when we achieved the lowest MAE values, and threshold correlation values were between 0.1 to 0.2. These threshold correlation values might not remain valid in another dataset. This experiment needs to be performed to determine the optimal threshold value for the filtering process before utilizing the threshold correlation value in the CSCCF model. If we can determine the threshold correlation value C_T, then the similarity calculation in our opinion prediction model will be updated in the following way:

$$Similarity(x,y) = Cosine\ Sim\ (U_x^{\wedge}, U_y^{\wedge})$$

$$= \frac{\sum_{i=1, i\neq t}^{n} \begin{cases} C_{ti}^2 R_i^x R_i^y & if\ C_{ti} \geq C_T \\ 0 & else \end{cases}}{\sqrt{\sum_{i=1,i\neq t}^{n} \begin{cases} C_{ti}^2 (R_i^X)^2 & if\ C_{ti} \geq C_T \\ 0 & else \end{cases}} + \sqrt{\sum_{i=1,i\neq t}^{n} \begin{cases} C_{ti}^2 (R_i^y)^2 & if\ C_{ti} \geq C_T \\ 0 & else \end{cases}}} \quad (11)$$

5.6 Opinion Prediction Model Application

CSCCF opinion prediction model can be used to analyze different social phenomena in our ICAS system. In this section, we analyzed Group Representation phenomena to showcase how the opinion prediction model can be used in our system.

5.6.1 Group Representation in the Discussion:

In our system, users contribute to the discussion by posting numerous arguments. The arguments generally contain the opinions, rationale, ideas, etc. favoring the participating user's opinion or perspective on the issue. On a collective level, the entire discussion content represents the viewpoint, opinions, rationale, etc. of the participating users. However, typically users with different perspectives do not participate in the discussion proportionally. If a particular opinionated group participates in the discussion mostly, then they will contribute to most of the discussion content. The overall tone of the arguments in the discussion might favor their opinion values. And if a particular opinionated user group does not participate in the discussion, the discussion would not represent their viewpoint at all. When a new user reads the discussion, he/she might get the idea that the majority of the people have this one particular kind of opinion on this issue as the majority of the arguments favors this viewpoint. However, this may not be the real scenario. Users with different perspectives other than the participated ones did not have significant enough participation in the discussion to be noticed or give people ideas about their opinions, ideas, viewpoint, etc. on the issue. This may create some bias to the reader's mind as the discussion content is not proportionally representative of different opinionated user groups. We can measure how much a particular opinionated group is represented in the discussion so that we can inform the readers how representative a discussion is. We will measure this phenomenon using the "Group Representation" metric.

To measure this group representation metric, we need to group users based on their opinion on an issue. Then for each group, we need to measure the percentage of the total users this group covers. We will also measure the portion of the discussion content each group contributed to the discussion. Using the user and argument coverage, we will measure the group representation value for each opinionated group in the discussion.

We defined the following term “User Coverage” to measure the portion of the whole user-space a particular group covers.

$$User\ Coverage = \frac{Number\ of\ Users\ in\ a\ Group}{Total\ Number\ of\ Users} \qquad (11)$$

We defined another term “Argument Coverage” to measure the portion of arguments in the discussion a particular group contributed to convey their idea, beliefs in the discussion.

$$Argument\ Coverage = \frac{Number\ of\ Arguments\ by\ a\ Group}{Total\ Number\ of\ Arguments} \qquad (12)$$

For a user group, if user coverage and argument coverage are equal, then this group is ideally represented in the discussion. If argument coverage is higher than user coverage, then this group is over-represented, if lower then under-represented in the discussion. We defined the group representation for a group using the following equation:

$$Group\ Representation = \frac{Argument\ Coverage}{User\ Coverage} \qquad (13)$$

From the above discussion, we can see that in order to analyze this "Group Representation” phenomena, we need to cluster or group users based on their opinion on an issue. However, users did not participate in all the positions of an issue in our argumentation platforms. So, the resulting dataset contains lots of missing information. Clustering algorithms struggle to analyze the dataset with missing values. Typically they discard the users with missing values which limits the user

analysis scope. Clustering algorithms impute the missing values with global values such as observed mean, median or the most frequent values. However, imputation with such global values often introduces several problems and the resulting groups often have very little meaningful information. We can also impute the missing values with our CSCCF opinion prediction model. In the following section we will cluster the users imputing the missing value with global values and with the predicted values from CSCCF, then we will analyze which process gives more meaning user groups. With the resulted user groups, we will examine "Group Representation" phenomena in the discussion.

5.6.2 Clustering users with Traditional Imputation Approach

We imputed the missing opinion values of users at different positions using the mean agreement value from all users in that position. Then, we applied the K-Means clustering algorithm to group users based on their opinion on this issue with a different number of clusters and evaluated the clustering quality with the Silhouette score.

Table 5.1 contains the clustering result in Guns on Campus issue. In this issue, we have position 0 (G1), position 1 (G2), position 2 (G3), and position 3 (G4), and the best clusters we got, when the number of clusters was defined at 5. The mean agreement value for G1, G2, G3, and G4 positions of gun issue are 0.20, 0.11, 0.12, -0.43, respectively. In general groups merged users with missing values and users with near missing values together and put them in one group. Group 0 is made of users with missing values and users with near missing values at G2 position. Users of Group 4 has either missing values or near missing values at G4 position. Group 1 is the largest group out of 5 groups; its users have missing values at G2, G3, and G4 positions, or their agreement values are near the missing values. On other issues, we also observe the same phenomena of grouping users with near and missing together.

Table 5. 1 Group Characteristics for Gun Issue using Column Mean as Missing Value

Group No	Group Size	G1 : Value	G2: Value	G3 : Value	G4 : Value
0	39	-0.42	0.11	0.37	0.52
1	119	0.27	0.11	0.12	-0.43
2	51	0.70	-0.55	-0.4	-0.75
3	39	0.86	0.24	-0.6	-0.8
4	60	-0.50	0.36	0.56	-0.43

We also tried imputing the missing values with median agreement value and the most frequent agreement value in a position. The result pattern is the same as imputing the missing values with mean agreement value. Clustering algorithms treat the users with missing values and users with near missing values in a similar way and put these users into one single group. If these users did not have missing values, they might not be in the same group. So, the output groups generated from the clustering algorithms are not reliable and contain miss grouping of many users.

5.6.3 Clustering users with Predicted Values from CSCCF

We imputed the missing values using our the CSCCF for each missing opinion values in the dataset. On the complete dataset, we then applied the K-Means clustering algorithms to group users based on their opinion within an issue. This clustering results we got this time are much improved and better than the three missing value imputation methods discussed in the earlier section. This time the clustering algorithm did not put the users with missing values at a position together into one group. Also, the output groups have definite characteristics than the previously generated opinionated user groups. The following table 5.2 contains the group results generated from the clustering algorithm with each group number, their size, overall group opinion (average agreement value) at four positions (Positions 0 (G1), Position 1 (G2), Position 2 (G3), and Position

4 (G4)). In the last row, it also shows the overall user opinion (average agreement values at four positions) of all users in the system.

Table 5. 2 Statistics of Different User Groups for Gun Issue

Group No	Group Size	G1 : Value	G2: Value	G3 : Value	G4 : Value
1	61	0.75	0.33	-0.2	-0.7
2	43	-0.53	0.29	0.38	0.40
3	71	-0.30	0.35	0.54	-0.44
4	47	0.75	-0.55	-0.6	-0.8
5	86	0.47	0.05	0.37	-0.55
overall	308	0.37	0.18	0.23	-0.55

Both Group no 1 and Group no 4 strongly supports that college campuses should not allow students to carry firearms under any circumstances. But Group no 1 does not hold this belief for special cases of allowing to carry concealed firearms by those who receive special permission. In contrast, Group no 4 does not favor for these special cases. However, both Group no 1 and Group no 4 disagree that a concealed carry permit or additional training would validate students to carry guns on campus. Group 3 is more approving for students carrying guns but with restrictions like special permission, additional training, or test than banning guns on campus totally or giving students full freedom to carry guns on campus. Group 2 has a similar opinion to group 3, but they support giving students' freedom to carry guns on campus. Group 5 is the largest of these five groups in user numbers; supports mostly that carry permit is not enough; some restrictions should be applied to allow students to carry firearms on campus.

From the above discussion, we have showed that how the prediction model helped us to identify user groups with definitive characteristics compared to the results from the previous missing value

imputation results. With these defined user groups, we analyzed user group representation in the various issue discussions.

5.6.4 Group Representation Experimental Results

Table no 5.3 contains the group representation results for five different groups in position 2 discussion. Group no 0, 4 is under-represented, but Group no 1, 2, and 3 are over-represented in the position 2 discussion. Group nos 4 is the largest group in user size but did not have the highest number of arguments in the discussion. Even though Group 2 was not the largest group user size-wise, according to the number of arguments, they are the largest group represented in the discussion. So, they are over-represented in the discussion.

Table 5. 3 Group Representation of Different User Groups for Gun Issue at position 2

Group No	Group Size	Group Percentage	Number of Arguments	Argument Percentage	Group Representation	Representation
0	61	0.198051948	97	0.167241379	0.844431882	Under-represented
1	43	0.13961039	85	0.146551724	1.049719326	Over-represented
2	71	0.230519481	153	0.263793103	1.144341914	Over-represented
3	47	0.152597403	106	0.182758621	1.197652238	Over-represented
4	86	0.279220779	139	0.239655172	0.85829992	Under-represented

From figure 5.10, we can see whether a group is over or under represented in the discussion at all four positions on gun issue. Group no 0 is under-represented in all four positions. Group no 1 is over-represented in all positions except position 1. Group no 2 is under-presented in position 0 and position 3, but over-represented in position 1 and position 2. Group no 3 is over-represented in all

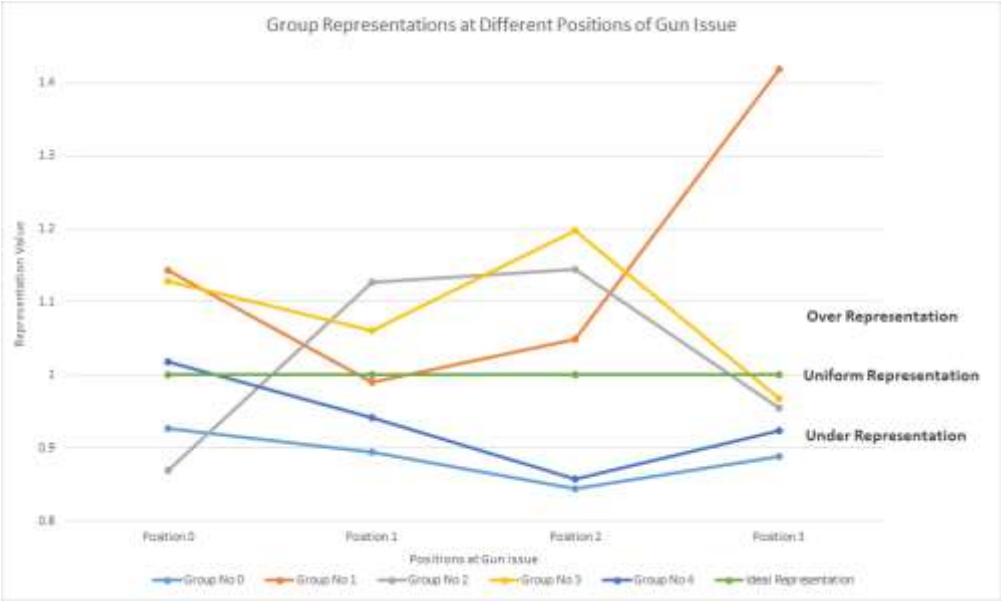

Figure 5. 10 Group Representation of different groups at different positions of Gun Issue

positions except position 3. Group no 4 is under-represented in all positions except position 1. From the results we can see that, Group no 1 is the most over-represented group in all the discussions. And Group no 0 is the most under-represented group in all the discussions.

5.7 Discussion

Our model CSCCF outperformed other comparison models in predicting opinion across issues. We think the main reason is because of people's similarity in terms of their values, as described by Schwartz theory of basic human values [20]. Political leanings on social issues such as conservative, liberal, moderate conservative, moderate liberal, etc. and also their stance on religion are few of the issues inferred from their values. In our system, different positions across issues are designed to capture certain opinionated perspectives or political leanings. Although these positions are in different issues, they are correlated in terms of their political leanings and perspectives. Generally, people gravitate towards a particular opinionated perspective on the issue based on their political leanings or their political party association such as democratic, republican, etc. Their perspectives across issues are generally consistent, and our model CSCCF captures this

information using the correlation between different positions across different issues to predict user opinion in a non-participated position of related issue.

The CSCCF has the limitation of data items being correlated with each other in some way. If there is a strong correlation between data items, then CSCCF would produce good accuracy. But if the data items are not correlated at all, CSCCF will filter out all data items and will not be able to make predictions. Determining the threshold correlation value, as described in the experiment section, would be a good idea before using the model to enforce validity and high accuracy. We think the CSCCF model can be a good fit for the scenario where there is a scarcity of available data to learn about the individual user, and the data items are globally correlated in some way. When the overall user data space is sparse, using a global correlation might help the prediction models to handle the data sparsity problem.

CSCCF Opinion prediction model can help us to achieve user groups with defined characteristics. Once we have defined user groups, we can use these user groups for different group related analytical models, group behavior, activity, and interaction within the group and with other groups in our argumentation platform. Analysis of these events will enable us to effectively analyze these phenomena and use the findings and teachings from these analyses into different models developed in our argumentation platform.

5.8 Related Work

5.8.1 Opinion Analysis on Argumentation Platform

Various research works focused on mining and analyzing user opinion from underlying discussion data in the cyber argumentation system. These works mostly focused on analyzing the impact of interaction with different opinionated people and how it affects their overall opinion, such as

Opinion space [7] and Considerit [21]. Some platforms such as Citizen report card [22], Open Town Hall [23], and California report card [24] focused on surveying collective user opinion on vital issues from a public service perspective. In these systems, users don't have a lot of ways of interacting with others, so there is less opportunity to exchange views and ideas effectively [25]. So, we used our interactive ICAS platform to analyze and predict user opinion. In addition, these platforms analyzed collective user opinion from actual user participation data only, none of these platforms predicted user opinion in the non-participated issues.

5.8.2 Opinion Prediction on Social Media

Social media data is often used to predict collective user attitudes or opinions. Researchers have crawled political discussion data to identify the users' political stance [26] or to predict a particular political outcome [27]. Researchers have also used social media data to predict user reactions on different social events, such as the 2015 Paris Terror Attack [28]. Many researchers used social media data to predict users' opinions on important issues/people using different algorithms (see [29] [8] [30] for examples). These works mostly looked at predicting an individual or group's opinion on a single issue using the related textual content on that issue only. One of the significant differences is that these works did not use user opinion in one or multiple issues to predict user opinion in another issue like our opinion prediction method presented in this paper.

In contrast to argumentation platform data, social media data are vast, noisy, unstructured, and dynamic in nature [31]. Often people use Natural Language Processing (NLP) on it to identify user opinion. However, ambiguity, implicit opinion expression, and domain-specific ideas makes NLP based approaches ineffective in many cases [32]. Our system allows users to explicitly state their agreement values, which enables us to mine user opinion from numerical agreement values avoiding the opinion extraction using NLP.

5.8.3 Multi-Issue Opinion Prediction

To our knowledge, little work has been done in opinion prediction across multiple issues. Probabilistic Matrix Factorization (PMF) approach is used to fill out a user-opinion matrix on different issues or topics [10]. However, this was an intermediate step of predicting the polarity of interaction between users, and the authors did not evaluate the accuracy of the prediction step. [9] used collaborative filtering to predict the user's opinion on important political topics, then the users were clustered into political parties. In a follow-up paper [33] they used topic distribution from user arguments, user interaction, and profile data to infer a user's stance on an issue. The model was based on the idea that users with similar topic distribution in their arguments will have a similar position on an issue. In their system, each issue only had two positions, and users can only agree or disagree with it. Whereas in our system, each issue can have multiple positions representing different viewpoints or solutions on the issue, and the user can agree or disagree with a position with a level of agreement with it. Their process includes topic modeling as a step; however, topic modeling is computationally expensive and requires predefined parameter tuning like the number of topics. Also, each time user adds a new argument, the topic distribution needs to be generated again. Our model does not require a computationally expensive operation to infer the user's updated agreement value.

5.8.4 Variations of Collaborative Filtering

A memory-based collaborative filtering algorithm calculates the similarity between users/ items from the whole or subset of the dataset and generates prediction from top similar neighbors. Similarity measurement between users/ items and predicting ratings from top similar ones are the two main ways these algorithms differ with each other. One of the significant differences between these collaborative filtering algorithms is how they calculate the similarity between users or items.

One popular approach measures the correlation value between two users or items from their associated historical data and use it as the similarity value between users or items [34]. The more correlated these values are, the more similar they are in these collaborative filtering based methods. Pearson Correlation, Spearman rank correlation, Kendall's tau correlation, Constrained Pearson Correlation are some of the examples of this approach. Another popular approach uses the user or item vector and measures the cosine similarity among them [34]. Researchers applied different modifications with these basic approaches. Some examples are the use of rank, mean, median etc. values instead of rating values [35], [36], emphasizing high weights and punishing low weights [37], the number of common rated items by users [38], whether the rated items are universally liked [39] etc. in the similarity calculation. Model-based collaborative filtering methods implement different clustering methods in CF [40], dimensionality reduction such as SVD, PCA based CF [41], [42], Bayesian belief net based CF [43] in the collaborative filtering mechanism. Hybrid collaborative filtering methods combine memory-based, model-based, or content boosted CF algorithms [44] together to improve the performance. According to our knowledge, no similarity method uses globally calculated correlation values of items as weight in cosine similarity measurement, like our method presented in this paper.

However, the correlation value has been used in collaborative filtering approaches, but mostly in between different data domains, and not within a single data domain. Some of the examples of these approaches are Collective Link Prediction, Multi-domain Collaborative Filtering [35]. These methods use different learning-based methods to exploit domain correlation. Correlation values are also used when an entity or user participates in multiple relations with different data items in the collective or relational matrix factorization method [46]. Cross-domain CF models also use this matrix factorization approach via a coordinate system transfer method [45]. Although these

methods use correlation, but they are computationally expensive and generally used for data items that vary in multiple domains, or user-data items correspond to numerous relations. And in our case, user-data items correspond to a single relationship and correlation is used within one data domain, and overall our method is computationally inexpensive compared to these models.

5.8.5 Clustering with Missing Values

Clustering algorithms generally struggle to analyze and find groups in a dataset with missing values. Typically, clustering algorithms handle missing values as a preprocessing step, either by ignoring data with missing values or filling missing values with imputed values. Missing values imputation is a common and challenging issue in data mining field. Popular approaches fill the missing values manually or replace them with global constant or mean of the object [47]. Observed mean, median values for the features are used to fill the missing values in the dataset [48]. Another approach is to model the distribution of the data and fill the missing values using the data distribution [48]. Missing values are also imputed from the closest matching patterns or other information of the pattern [49]. Regression-based imputation uses the predicted values from a regression analysis [49]. The similarity of users or items in the data is also used to predict the missing values [50]. Different Neural net, probabilistic models, collaborative filtering, and matrix factorization based approaches have also been used to impute the missing values. Marginalization approach ignores the data with missing values, but that limits the analysis scope [47].

5.9 Conclusion

In this research work, we developed a multi-issue opinion prediction method for large scale cyber argumentation platform. Our method predicts how much a user would agree/ disagree with a particular position on an issue using similar user and opinion correlations between different

positions on related issues. To our best knowledge, this is the first attempt to estimate partial user agreement in a cyber-argumentation platform. With different experiment analysis, we evaluated the accuracy of our model and compared with other baseline predictions methods. Our model achieved high accuracy and outperformed other baseline models with a Mean Absolute Error(MAE) value of 0.133. In this work, we also evaluated different scenarios that can impact the model's prediction accuracy, such as the number of positions being predicted, degree of correlation, performance on smaller training data, etc. As our model exploits correlation values, we evaluated the performance of comparison models on the preprocessed and filtered dataset by different correlation degrees to demonstrate that the CSCCF model uses the opinion correlation in a better way than other comparison models. In this work, we also analyzed group-representation phenomena to demonstrate how our CSCCF opinion prediction model can be used in our system. Our method exploits the correlation values of different issues being discussed on the platform to achieve high accuracy, so if the issues are not correlated at all, our model will not work. How related different issues are in the discussion need to be considered before using the CSCCF model. In a cyber-argumentation environment, our model can be used to estimate user's opinions with high accuracy on related issues on which they did not express their opinion explicitly. The predicted opinion values can also help to assess different collective intelligence more effectively, especially when the user participation is incomplete in a multi-issue cyber argumentation platform.

5.10 References

[1] P.M. Dung, On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games, Artificial Intelligence 77(2) (1995), 321–357. doi:10.1016/0004-3702(94)00041-X. http://www.sciencedirect.com/science/article/pii/000437029400041X.

[2] W. Kunz and H.W.J. Rittel, Issues as elements of information systems (1970). https://trove.nla.gov.au/version/21020901.

[3] S.E. Toulmin, The Uses of Argument by Stephen E. Toulmin, 2003. doi:10.1017/CBO9780511840005. /core/books/ uses-of-argument/26CF801BC12004587B66778297D5567C.

[4] M.J. Klein, The CATALYST Deliberation Analytics Server, 2015. doi:10.2139/ssrn.2962524.

[5] J. Sirrianni, X. Liu and D. Adams, Quantitative Modeling of Polarization in Online Intelligent Argumentation and Deliberation for Capturing Collective Intelligence, 2018 IEEE International Conference on Cognitive Computing (ICCC) (2018), 57–64. doi:10.1109/ICCC.2018.00015.

[6] R.S. Arvapally and X.F. Liu, Analyzing credibility of arguments in a web-based intelligent argumentation system for collective decision support based on K-means clustering algorithm, Knowledge Management Research & Practice 10(4) (2012), 326–341. doi:10.1057/kmrp.2012.26.

[7] S. Faridani, E. Bitton, K. Ryokai and K. Goldberg, Opinion Space: A Scalable Tool for Browsing Online Comments, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '10, ACM, New York, NY, USA, 2010, pp. 1175–1184. ISBN 978-1-60558-929-9. doi:10.1145/1753326.1753502.

[8] O. Fraisier, G. Cabanac, Y. Pitarch, R. Besançon and M. Boughanem, Stance Classification Through Proximity-based Community Detection, in: Proceedings of the 29th on Hypertext and Social Media, HT '18, ACM, New York, NY, USA, 2018, pp. 220–228. ISBN 978-1-4503-5427-1. doi:10.1145/3209542.3209549.

[9] S. Gottipati, M. Qiu, L. Yang, F. Zhu and J. Jiang, Predicting User's Political Party Using Ideological Stances, in: Social Informatics, A. Jatowt, E.-P. Lim, Y. Ding, A. Miura, T. Tezuka, G. Dias, K. Tanaka, A. Flanagin and B.T. Dai, eds, Lecture Notes in Computer Science, Springer International Publishing, 2013, pp. 177–191. ISBN 978-3-319-03260-3.

[10] M. QIU, Mining user viewpoints in online discussions, Dissertations and Theses Collection (Open Access) (2015), 1–119. https://ink.library.smu.edu.sg/etd_coll/127.

[11] H. Tajfel and J.C. Turner, The Social Identity Theory of Intergroup Behavior, in: Political Psychology, 0 edn, J.T. Jost and J. Sidanius, eds, Psychology Press, 2004, pp. 276–293. ISBN 978-0-203-50598-4. doi:10.4324/9780203505984-16.

[12] X. Liu, E. Khudkhudia, L. Wen and V. Sajja, An Intelligent Computational Argumentation System for Supporting Collaborative Software Development Decision Making, 2010, pp. 167–180. doi:10.4018/978-1-60566-758-4.ch009.

[13] S. Sigman and X.F. Liu, A computational argumentation methodology for capturing and analyzing design rationale arising from multiple perspectives, Information & Software Technology 45 (2003), 113–122. doi:10.1016/S0950-5849(02)00187-8.

[14] X.F. Liu, S. Raorane and M.C. Leu, A Web-based Intelligent Collaborative System for Engineering Design, in: Collaborative Product Design and Manufacturing Methodologies and Applications, W.D. Li, C. McMahon, S.K. Ong and A.Y.C. Nee, eds, Springer Series in Advanced Manufacturing, Springer London, London, 2007, pp. 37–58. ISBN 978-1-84628-802-9. doi:10.1007/978-1-84628-802-9_2.

[15] X.F. Liu, E.C. Barnes and J.E. Savolainen, Conflict Detection and Resolution for Product Line Design in a Collaborative Decision Making Environment, in: Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work, CSCW '12, ACM, New York, NY, USA, 2012, pp. 1327–1336, event-place: Seattle, Washington, USA. ISBN 978-1-4503-1086-4. doi:10.1145/2145204.2145402.

[16] R.S. Arvapally and X.F. Liu, Polarisation assessment in an intelligent argumentation system using fuzzy clustering algorithm for collaborative decision support, Argument & Computation 4(3) (2013), 181–208. doi:10.1080/19462166.2013.794163. https://content.iospress.com/articles/argument-and-computation/794163.

[17] M.R. Smith, M.S. Gashler and T. Martinez, A hybrid latent variable neural network model for item recommendation, in: 2015 International Joint Conference on Neural Networks (IJCNN), 2015, pp. 1–7. doi:10.1109/IJCNN.2015.7280324.

[18] G. Takács, I. Pilászy, B. Németh and D. Tikk, Matrix Factorization and Neighbor Based Algorithms for the Netflix Prize Problem, in: Proceedings of the 2008 ACM Conference on Recommender Systems, RecSys '08, ACM, New York, NY, USA, 2008, pp. 267–274, event-place: Lausanne, Switzerland. ISBN 978-1-60558-093-7. doi:10.1145/1454008.1454049.

[19] A. Mnih and R.R. Salakhutdinov, Probabilistic Matrix Factorization, in: Advances in Neural Information Processing Systems 20, J.C. Platt, D. Koller, Y. Singer and S.T. Roweis, eds, Curran Associates, Inc., 2008, pp. 1257–1264. http://papers.nips.cc/paper/3208-probabilistic-matrix-factorization.pdf.

[20] S. Schwartz, An Overview of the Schwartz Theory of Basic Values, Online Readings in Psychology and Culture 2(1) (2012). doi:10.9707/2307-0919.1116. https://scholarworks.gvsu.edu/orpc/vol2/iss1/11.

[21] T. Kriplean, J. Morgan, D. Freelon, A. Borning and L. Bennett, Supporting Reflective Public Thought with Considerit, in: Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work, CSCW '12, ACM, New York, NY, USA, 2012, pp. 265–274. ISBN 978-1-4503-1086-4. doi:10.1145/2145204.2145249.

[22] Participation & Civic Engagement - Citizen Report Card and Community Score Card. http://web.worldbank.org/WBSITE/EXTERNAL/TOPICS/EXTSOCIALDEVELOPMENT/EXTPCENG/0,,contentMDK:20507680~pagePK:148956~piPK:216618~theSitePK:410306,00.html.

[23] The Extent of Public Participation. https://icma.org/articles/pm-magazine/extent-public-participation.

[24] Civic Media Project: The California Report Card Version 1.0. http://civicmediaproject.org/works/civic-media-project/thecaliforniareportcard.

[25] M. Nelimarkka, B. Nonnecke, S. Krishnan, T. Aitumurto, D. Catterson, C. Crittenden, C. Garland, C. Gregory, C.-C.A. Huang, G. Newsom, J. Patel, J. Scott and K. Goldberg, Comparing Three Online Civic Engagement Platforms using the Spectrum of Public Participation., in: THE INTERNET, POLICY & POLITICS CONFERENCES, Oxford, 2014. https://escholarship.org/uc/item/0bz755bj.

[26] M. Boireau, Determining Political Stances from Twitter Timelines: The Belgian Parliament Case, in: Proceedings of the 2014 Conference on Electronic Governance and Open Society: Challenges in Eurasia, EGOSE '14, ACM, New York, NY, USA, 2014, pp. 145–151. ISBN 978-1-4503-3401-3. doi:10.1145/2729104.2729114.

[27] A. Tumasjan, T.O. Sprenger, P.G. Sandner and I.M.Welpe, Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment, in: Fourth International AAAI Conference on Weblogs and Social Media, 2010. https://www.aaai.org/ocs/index.php/ICWSM/ICWSM10/paper/view/1441.

[28] W. Magdy, K. Darwish, N. Abokhodair, A. Rahimi and T. Baldwin, #ISISisNotIslam or #DeportAllMuslims?: Predicting Unspoken Views, in: Proceedings of the 8th ACM Conference on Web Science, WebSci '16, ACM, New York, NY, USA, 2016, pp. 95–106. ISBN 978-1-4503-4208-7. doi:10.1145/2908131.2908150.

[29] D. Sridhar, L. Getoor and M.A. Walker, Collective Stance Classification of Posts in Online Debate Forums, in: Proceedings of Joint Workshop on Social Dynamics and Personal Attributes in Social Media, 2014, pp. 109–117.

[30] S.M. Mohammad, P. Sobhani and S. Kiritchenko, Stance and Sentiment in Tweets (2016). https://arxiv.org/abs/1605.01655v1.

[31] P. Gundecha and H. Liu, Mining Social Media: A Brief Introduction, in: New Directions in Informatics, Optimization, Logistics, and Production, INFORMS TutORials in Operations Research, INFORMS, 2012, pp. 1–17. ISBN 978-0-9843378-3-5. doi:10.1287/educ.1120.0105.

[32] E. Cambria, C. Havasi and A. Hussain, SenticNet 2: A Semantic and Affective Resource for Opinion Mining and Sentiment Analysis, in: 25th International Florida Artificial Intelligence Research Society Conference, 2012, pp. 202–207.

[33] M. Qiu, Y. Sim, N.A. Smith and J. Jiang, Modeling User Arguments, Interactions, and Attributes for Stance Prediction in Online Debate Forums, in: Proceedings of the 2015 SIAM International Conference on Data Mining, Society for Industrial and Applied Mathematics, 2015, pp. 855–863. ISBN 978-1-61197-401-0. doi:10.1137/1.9781611974010.96.

[34] X. Su and T.M. Khoshgoftaar, A Survey of Collaborative Filtering Techniques, Adv. in Artif. Intell. 2009 (2009), 4:2–4:2. doi:10.1155/2009/421425.

[35] K. Goldberg, T. Roeder, D. Gupta and C. Perkins, Eigentaste: A Constant Time Collaborative Filtering Algorithm, Information Retrieval 4(2) (2001), 133–151. doi:10.1023/A:1011419012209.

[36] J.L. Herlocker, J.A. Konstan, L.G. Terveen and J.T. Riedl, Evaluating collaborative filtering recommender systems, Acm Transactions on Information Systems 22 (2004), 5–53.

[37] J.S. Breese, D. Heckerman and C. Kadie, Empirical Analysis of Predictive Algorithms for Collaborative Filtering, in: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, UAI'98, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998, pp. 43–52, event-place: Madison, Wisconsin. ISBN 978-1-55860-555-8. http://dl.acm.org/citation.cfm?id=2074094.2074100.

[38] H. Liu, Z. Hu, A. Mian, H. Tian and X. Zhu, A New User Similarity Model to Improve the Accuracy of Collaborative Filtering, Know.-Based Syst. 56(C) (2014), 156–166. doi:10.1016/j.knosys.2013.11.006.

[39] G. Salton and M.J. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, Inc., New York, NY, USA, 1986. ISBN 978-0-07-054484-0.

[40] J. Han, M. Kamber and J. Pei, Data Mining: Concepts and Techniques, 3rd edn, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2011. ISBN 978-0-12-381479-1.

[41] D. Billsus and M.J. Pazzani, Learning Collaborative Information Filters, in: Proceedings of the Fifteenth International Conference on Machine Learning, ICML '98, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998, pp. 46–54. ISBN 978-1-55860-556-5. http://dl.acm.org/citation.cfm?id=645527.657311.

[42] K. Pearson, LIII.On lines and planes of closest fit to systems of points in space, Philosophical Magazine Series 6 (1901). doi:10.1080/14786440109462720. https://www.scienceopen.com/document?vid=d9332914-ac23-498a-b2db-1c3e8964c146.

[43] X. Su and T.M. Khoshgoftaar, Collaborative Filtering for Multi-class Data Using Belief Nets Algorithms, in: 2006 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06), 2006, pp. 497–504. doi:10.1109/ICTAI.2006.41.

[44] P. Melville, R.J. Mooney and R. Nagarajan, Content-boosted Collaborative Filtering for Improved Recommendations, in: Eighteenth National Conference on Artificial Intelligence, American Association for Artificial Intelligence, Menlo Park, CA, USA, 2002, pp. 187–192, event-place: Edmonton, Alberta, Canada. ISBN 978-0-262-51129-2. http://dl.acm.org/citation.cfm?id=777092.777124.

[45] A. Li, Cross-Domain Collaborative Filtering: A Brief Survey, in: 2011 IEEE 23rd International Conference on Tools with Artificial Intelligence, 2011, pp. 1085–1086. doi:10.1109/ICTAI.2011.184.

[46] A.P. Singh and G.J. Gordon, Relational Learning via Collective Matrix Factorization, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '08, ACM, New York, NY, USA, 2008, pp. 650–658, event-place: Las Vegas, Nevada, USA. ISBN 978-1-60558-193-4. doi:10.1145/1401890.1401969.

[47] S. Zhang, J. Zhang, X. Zhu, Y. Qin and C. Zhang, Transactions on Computational Science I, M.L. Gavrilova and C.J.K. Tan, eds, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 128–138, Chapter Missing Value Imputation Based on Data Clustering. ISBN 3-540-79298-8, 978-3-540-79298-7. http://dl.acm.org/citation.cfm?id=1805820.1805829.

[48] K.Wagstaff, Clustering with Missing Values: No Imputation Required, Chicago, IL, United States, 2004. https://ntrs.nasa.gov/search.jsp?R=20070019774.

[49] M. Sarkar and T.Y. Leong, Fuzzy K-means clustering with missing values., Proceedings of the AMIA Symposium (2001), 588–592. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2243620/.

[50] Z. Ghahramani and M.I. Jordan, Learning from Incomplete Data, Technical Report, Massachusetts Institute of Technology, Cambridge, MA, USA, 1994.

Chapter 6: Opinionated Group Detection and Demographics Analysis under Different Issues in Cyber-Argumentation Platform

6.1 Abstract

In the Cyber-Argumentation platform, participants discuss different important issues in the discussion. Groups can be implicit in the cyber-argumentation environment, and individuals from different groups can interact un-coordinately. In this chapter, I discussed the group's identification process from the discussion. Political leanings and social profile information are also described in this chapter. These groups were used in different research works, which are described in chapters 7 and 8.

6.2 Group Detection in Discussion

We grouped users based on their opinion on an issue using a clustering algorithm. However, users did not participate in all the positions of an issue, so the dataset contains lots of missing values. Clustering with missing values introduces error and bias in the clustered groups and any analytical results based on the derived groups [1]. This is why we used Cosine Similarity with position Correlation Collaborative Filter (CSCCF) opinion prediction model [2] to predict the missing opinion values. CSCCF [2] is a collaborative filtering based model which predicts user's opinion in an issue using similar users' opinion on related issues. With the CSCCF model, we generated a complete user-opinion dataset. Then for each issue, we applied the K-Means clustering algorithm to group users based on their opinion on the issue. We tested with a different number of clusters and selected the best one based on the Silhouette score value of clusters.

6.3 Group Information

Table 6. 1Group Analytics on the issue of "Guns on Campus"

Group No	Number of Members	Group Category	Percentage of total Male Population	Percentage of total Female population	Percentage of total White population	Percentage of total non-white population
0	61	Overall Liberal Group	14.93%	20.40%	18.11%	18.57%
1	43	Most Conservative Group	20.15%	7.96%	14.72%	5.71%
2	55	Overall Conservative Group	27.61%	16.92%	21.13%	21.43%
3	47	Most Liberal Group	8.96%	17.41%	11.70%	22.86%
4	86	Overall in the Middle	20.15%	29.35%	26.42%	22.86%
Total			100%	100%	100%	100%

We labeled the political leanings of the group based on their opinions in different positions in the issue. All the positions were previously designed to capture different perspectives of political beliefs such as conservative, lean conservative, lean liberal, liberal, etc. on the issue. We ranked the groups manually based on their support and opposition level to the conservative or liberal positions and attached a tag as their political leaning. These tag values are the most conservative, most liberal, overall liberal, overall conservative, and overall in the middle. We also analyzed the social profile information of the group members. Out of all social profile attributes, Gender, and Race had some significant pattern across the groups. On the Gender feature, we calculated what percentage of total Male and Female population each group covers. On the Race feature, we calculated the percentage of the total White/Non-White population each group covers. The following sub-sections contain the summarized political leanings and social profile information of different groups in the issues.

6.3.1 Group Analytics on the issue of "Guns on Campus"

The summary of the group Analytics on the Issue of "Guns on Campus" is presented in Table 6.1. Female users are mostly in the overall in the middle group or Liberal groups. Only a small percentage (7.96%) female users are in the most conservative group. Most of the male populations are in the conservative group. Only a small percentage (8.96%) of the total male populations are in the most liberal group.

6.3.2 Group Analytics on the Issue of "Same Sex Couples and Adoption"

The summary of the group Analytics on the Issue of "Same Sex Couples and Adoption" is presented in Table 6.2. Most of the populations are in the liberal groups. Only 7.96% of female, and 15.67% of male populations are in conservative groups. Only 10.19% of white, and 14.29% of non-white populations are in conservative groups.

Table 6. 2 Group Analytics on the Issue of "Same Sex Couples and Adoption"

Group No	Group Total	Group Category	Percentage of total Male Population	Percentage of total Female population	Percentage of total White population	Percentage of total non-white population
0	37	Conservative	15.67%	7.96%	10.19%	14.29%
1	95	Most Liberal One	25.37%	30.35%	26.79%	34.29%
2	110	Liberal	29.10%	35.32%	34.72%	25.71%
3	66	Liberal	21.64%	18.41%	20.38%	17.14%
Total			100%	100%	100%	100%

6.3.3 Group Analytics on the Issue of "Government and Healthcare"

The summary of the group Analytics on the Issue of "Government and Healthcare" is presented in Table 6.3. Most of the populations are on conservative groups. Only 8.21% of male, 8.96% of female population are in liberal groups. Only 6.04% of white, and 18.57% of non-white population are in liberal groups.

Table 6. 3 Group Analytics on the Issue of "Government and Healthcare"

Group No	Group Total	Group Category	Percentage of total Male Population	Percentage of total Female population	Percentage of total White population	Percentage of total non-white population
0	117	Conservative	35.07%	34.83%	33.96%	38.57%
1	50	Conservative	17.16%	13.43%	15.85%	11.43%
2	42	Most Conservative	13.43%	11.94%	13.96%	7.14%
3	70	Overall in the Middle	17.91%	22.89%	22.26%	15.71%
4	29	Liberal	8.21%	8.96%	6.04%	18.57%
Total			100%	100%	100%	100%

6.3.4 Group Analytics on the Issue of "Religion and Medicine"

The summary of the group Analytics on the Issue of "Religion and Medicine" is presented in Table 6.4. Male, Female, White, Non-white populations are pretty spread out across conservative and liberal groups.

Table 6. 4 Group Analytics on the Issue of "Religion and Medicine"

Group No	Group Total	Group Category	Percentage of total Male Population	Percentage of total Female population	Percentage of total White population	Percentage of total non-white population
0	52	Overall in the Middle	15.67%	15.42%	15.85%	14.29%
1	51	Liberal	19.40%	12.44%	14.72%	17.14%
2	77	Liberal	20.15%	24.88%	23.40%	21.43%
3	38	Conservative	10.45%	11.94%	12.08%	8.57%
4	41	Most Liberal Group	11.94%	12.44%	9.81%	21.43%
5	49	Liberal	14.18%	14.93%	16.23%	8.57%

6.4 Reference

[1] Shichao Zhang, Jilian Zhang, Xiaofeng Zhu, Yongsong Qin, and Chengqi Zhang. 2008. Missing Value Imputation Based on Data Clustering. In Transactions on Computational Science I, Marina L. Gavrilova and C. J. Kenneth Tan (eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 128–138. DOI:https://doi.org/10.1007/978-3-540-79299-4_7

[2] Md Mahfuzer Rahman, Joseph W. Sirrianni, Xiaoqing (Frank) Liu, and Douglas Adams. 2019. Predicting opinions across multiple issues in large scale cyber argumentation using collaborative filtering and viewpoint correlation. In The Ninth International Conference on Social Media Technologies, Communication, and Informatics (SOTICS 2019).

Chapter 7: Analysis and Modeling of Intra-group and Inter-group Interactions for Cyber Argumentation Platform

7.1 Abstract

In many cyber-argumentation platforms, people discuss important issues and interact with each other while supporting or criticizing each other's opinions in the discussion. These platforms focus more on intellectual debate and discussion and less on groups and communities among individuals. However, groups can be implicit in the cyber-argumentation environment, and individuals from these groups can interact with each other un-coordinately in the discussion. These actions can impact the participating individuals' opinions, influence their behavior, and can also affect the overall outcome of the discussion. Currently, there is no group interaction model developed for cyber-argumentation platforms, which can help us understand the impact of these group interactions at the individual and collective level in this kind of environment. To address these issues, we developed models that quantify different aspects of implicit group interaction in the cyber argumentation platform. Our first model quantifies the intra-group interactions between members of a group, which can help us understand how supportive or critical the members of a group are to each other. Our second model quantifies the inter-group interaction to analyze how supportive or critical the groups are to each other in the discussions. These two models consider the opinions of group members and the support/attack interaction pattern to other participants both inside or outside of their groups in the discussions to quantify intra-group and inter-group interactions in cyber argumentation.

7.2 Intra-Group Interaction Analysis

We developed an intra-group interaction model, which can be used to quantify the group interactions between members of a group. This model will analyze how supportive or critical the members of a group are to each other.

7.2.1 Intra-Group Interaction Graph

To analyze the intra-group interaction, we developed an intra-group interaction graph. In this graph, each node represents a user in the group, and each edge represents how supportive or critical it is $user_i$ to $user_j$. However, a user can interact with another user more than once. An average support-attack value between two members of a group can be calculated from all support-attack interactions between these two users. This scenario is depicted in Figure 1, where $user_i$ supported three times $user_j$ with +0.8, +0.9, and +0.8 agreement values, respectively. So, $user_i$ supported $user_j$ with +0.83 agreement value on average. In this way, we can calculate average support-attack values between every user pair in a group and generate the intra-group interaction graph. A sample intra-group user interaction graph is presented in figure 7.2. In this graph, edge values are calculated using an average support-attack value between users.

7.2.2 In-Group Support-Attack Degree

In-Group Support-Attack Degree analyzes how supportive or critical the group members are to each other within the group. Following are two basic approaches which can be used to quantify the in-group support-attack degree:

a) Normalized by Group Members

This approach averages support-attack values to calculate the support-attack value between two members of the group. Then, it calculates average support and attack values within the group using

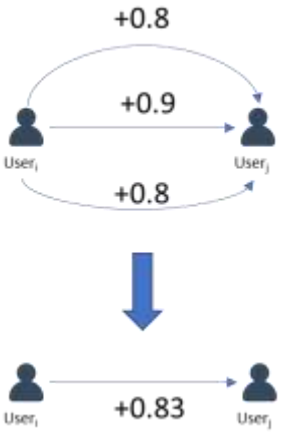

Figure 7. 2: Support-Attack value calculation between members in a group.

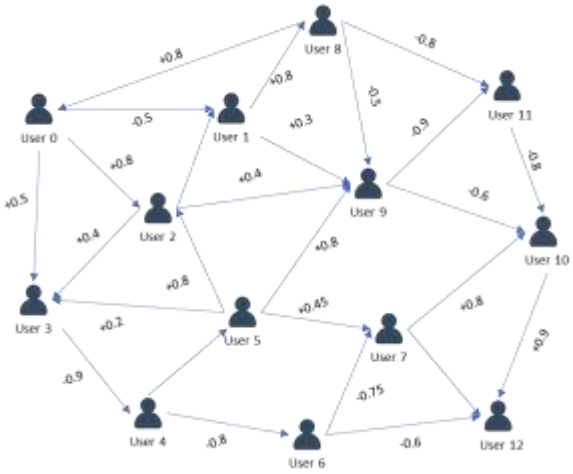

Figure 7. 1: Intra-group user interaction graph

average support and attack values between two group members. The in-group support-attack degree is calculated using the following way:

Support-Attack Degree in a Group =

$$\frac{Number\ of\ People\ Supported}{Total\ Number\ of\ People} \times Average\ Support\ Value\ +$$

$$\frac{Number\ of\ People\ Attacked}{Total\ Number\ of\ People} \times Average\ Attack\ Value$$

Every individual in a group has a similar priority in the group support - attack degree measurement, even if some members interact more within the group than other members.

b) Normalized by Interactions

This approach does not calculate the average support between two members of a group. Instead, it considers the number of support and attack interactions between the group members and calculates the support-attack degree in the following way:

Support-Attack Degree in a Group =

$$\frac{Number\ of\ Support\ Interaction}{Total\ Number\ of\ Interactions} \times Average\ Support\ Value\ +$$

$$\frac{Number\ of\ Attack\ Interactions}{Total\ Number\ of\ Interactions} \times Average\ Attack\ Value$$

7.2.3 Analytical Result

In-group support degree can be used to analyze members of which group are the most supportive or critical to each other in the discussion. Figure 7.3 contains an example of an in-group support degree calculated using two different approaches of different groups in the discussion of 'Guns of Campus' issue. We can see that Group 3 is the most supportive and Group 2 least supportive to its members in the discussion. Overall, group members are supportive of each other across all the groups.

7.3 Inter-Group Interaction Analysis

In our cyber-argumentation platform, users from different groups interact with each other via support/attack in discussions. The inter-group interaction model quantifies the group interaction between different groups. This model analyzes how supportive or critical the groups are to each other in the discussions.

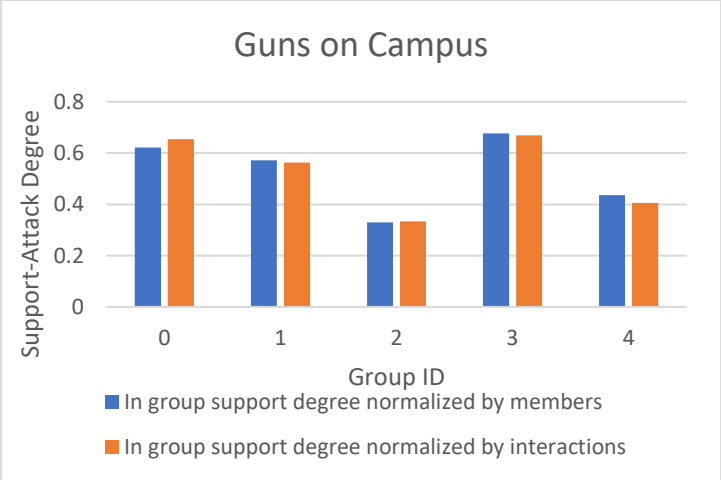

Figure 7. 3 In-Group Support-Attack Degree in the discussion of 'Guns on Campus' issue

7.3.1 Support-Attack Degree between Groups

This metric analyzes the support/attack pattern form the members of one group to the members of another group and quantifies how supportive or critical is one group to another group. Following is a fundamental approach to show how the support-attack degree between Group 0 to Group 1 can be calculated:

$$\textit{Support-Attack Degree } (Grp0 \textit{ to } Grp\ 1) =$$

$$\frac{Number\ of\ Support\ Interaction\ (Grp1\ to\ Grp\ 2)}{Total\ Number\ of\ Interactions} \times Average\ Support\ Value\ (Grp0\ to\ Grp1) +$$

$$\frac{Number\ of\ Attack\ Interactions(Grp1\ to\ Grp\ 2)}{Total\ Number\ of\ Interactions} \times Average\ Attack\ Value(Grp0\ to\ Grp\ 1)$$

Figure 7.4 contains an example of how the support-attack degree between two groups can be calculated. In this example, three users of group 0 attacked three users of group 1, and on average group 0 attacked group 1 with 0.7 intensity.

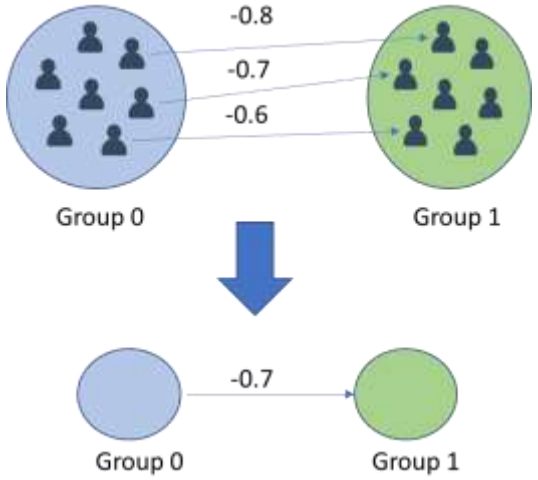

Figure 7. 4 : Support-Attack degree between group 0 to group 1

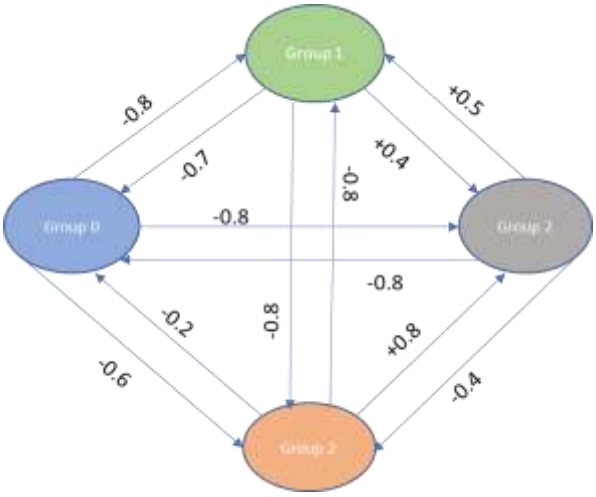

Figure 7. 5 : Inter-group interaction graph

7.3.2 Inter Group Interaction Graph

We built a directional inter-group interaction graph with the Support-Attack degree between different groups. In this graph, each group is represented by a node, and an edge represents the support degree between two groups. Figure 7.5 contains an example inter-group interaction graph between four different groups.

7.3.3 Inter-Group Interaction Analysis

Using the inter-group interaction graph, we can analyze group related phenomena and identify different characteristic groups. For example, we can determine which groups are overall supportive of other groups and which groups are critical to other groups. Figure 7.6 contains a sample of group support levels in the discussion of the "Religion and Medicine" issue. From this analysis, we can see that Group 2 is overall supportive of other groups, especially to groups 0 and 1. And groups 4 and 5 are the most supportive of group 2, although they did not receive much support from group 2. Figure 7.7 contains an example of a critical group in the discussion of the "Guns on Campus" issue. From this analysis, we can see that Group 1 is overall critical to other groups except for group 2. However, Group 1 did not receive any support from any other group.

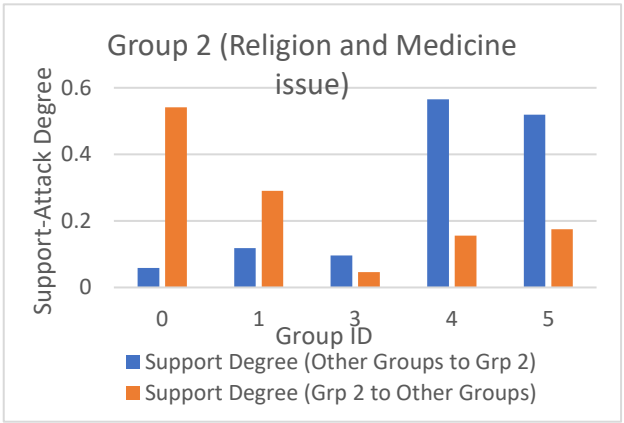

Figure 7. 5 Group 2 support analysis in Religion and Medicine Issue

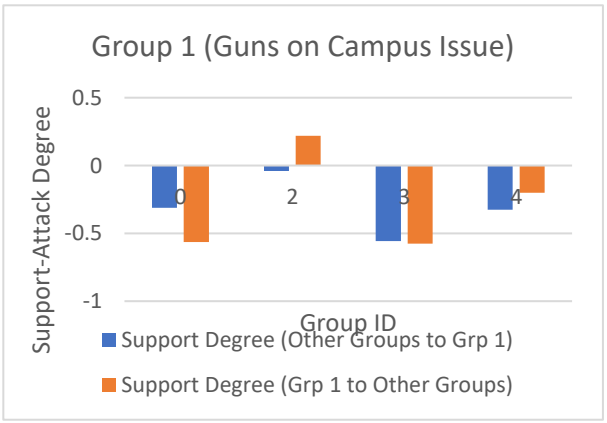

Figure 7. 4 Group 1 criticism analysis in Guns on Campus issue

Chapter 8: Group Identity Analysis utilizing Social Science Theories for User Behavior Analysis and Modeling in Cyber Argumentation Platform

8.1 Abstract

Many social and psychological theories have been used extensively in the design and implementation of various models and algorithms developed for user behavior modeling in web and social media platforms. Social Identity Theory (SIT) and Social Identity Model of Deindividuation Effects (SIDE) are two prominent theories, which explain the concept of self and the effects of anonymity on user behavior and activity when people interact with each other in a group environment. While these theories have been tested and validated in offline context or face-to-face interactions, there has been no validation for these theories in a cyber-argumentation setting. Cyber argumentation focuses more on intellectual debate and discussion and less on social groups and communities. Thus, it is not clear how much these theories hold in the cyber argumentation platform. In this paper, we designed an experiment to quantify the idea of group identity using Social Identity Theory and used it to inspect whether the key concepts of SIDE are valid in our argumentation platform. These key concepts analyze the impact of group identity and anonymity in user activity. To our knowledge, this is the first experiment, which examined the validity of different key concepts of SIDE before applying the findings in various models for web and cyber-argumentation platform.

8.2 Introduction

With the rise of the internet, people spend a significant amount of time using social media platforms. On these platforms, users not only seek information, but also interact with other individuals, make friends, discuss issues with other opinionated people from different

backgrounds, and collaborate with others. Just like face-to-face interactions, these online interactions shape an individual's personality and influence their behavior, opinions, attitudes, and beliefs. These online interactions lead to various social and psychological phenomena to occur in these environments. Researchers have found various examples and developed models to characterize several of these phenomena, including Groupthink [1], Polarized discussions [2], Group Polarization [3], etc.

Usually, researchers use different machine learning techniques and other computational approaches to model user behavior and measure different social phenomena in their online platforms. However, many social and psychological experiments have studied and developed theories and analytical models on this user behavior and phenomena in real life face-to-face settings. Often researchers incorporate the key concepts from these theories into their algorithms and system designs for the online environment without any evaluation of these theories [4, 5, 6, 7]. Even though these theories are valid in real-life face-to-face environments, they might not remain valid over computer-mediated communication in an online platform.

The Social Identity Theory (SIT) [8] and Social Identity Model of Deindividuation Effects (SIDE) [9] are two of the prominent theories, which analyze how people perceive themselves as group members and the effect of anonymous communications on people's behavior in a group environment. Social Identity Theory (SIT) [8] states that individuals develop a sense of groupness or group identity based on their perceived membership to a certain group in a social environment. Whereas the Social Identity Model of Deindividuation Effects (SIDE) states that anonymity changes the importance of group identity over personal identity among individuals, which influences their behavior to be more similar or dissimilar with the group they belong to.

The key concepts from these two theories might be useful in designing user behavior and group interaction models in online platforms. For example, SIT, and SIDE state that a person's opinion is more influenced by members of their in-group [8] than the members of their out-group [8]. The stronger a person identifies with their in-group [8], the more the group influences them. This insight can help in opinion prediction models when users interact with each other in an online group environment. However, before using the concepts from these theories in user opinion modeling and other social phenomena analysis models, we need to examine whether these theories are valid in an online discussion setting. To our knowledge, no one has reviewed the validity of SIT and SIDE in an online discussion environment before.

In this work, we examined the following three key concepts from SIDE in our argumentation platform using this group sense concept from SIT, a similar opinion on social issues [10].

Concept 1: Anonymity increases the importance of group identity among individuals in a group interaction environment.

Concept 2: Individuals with a strong group identity will be influenced to be more similar in their activity with their group.

Concept 3: Weak group identity does not influence user behavior so that user behavior will be more dissimilar with their group.

We used our argumentation platform, the Intelligent Cyber Argumentation System (ICAS), in which user participates in different issue discussions anonymously. We conducted an empirical study and collected a large dataset of 344 users discussing four important issues. We quantified the idea of "group sense" and "User Activity Similarity with Group" among individuals. With these ideas and collected empirical data, we analyzed whether the anonymity and group sense from a similar opinion is influencing users' behavior related to their in-group [8] and out-group [8]

activity as per SIDE in our platform. Our results show that the anonymity in our platform is triggering a strong group sense in the majority of the users. Besides, the majority of the users with strong group sense has similar activity with the group, which is symmetrical with the concepts from the SIDE. Only a marginal amount of users' activity is not being influenced by their group sense, as SIDE stated. These analytical results can help to analyze an individual's decision-making process and many argumentation phenomena such as collaborative opinions and decision making, in-group bias, expression of divergent viewpoint, groupthink, etc. in our ICAS platform. In this paper, we make the following contributions.

- We designed and performed an experiment to examine the validity of three key concepts of SIDE using concepts from SIT with the argumentation data collected via our platform.
- We performed a statistical significance test to verify that our observations on the group identity and group influence is not random.
- Our results demonstrate that in our platform
    - 73.5% of the total users have strong group sense.
    - 69.9% of the users with strong group sense also have similar strong activity with their group.
    - As per the validity of SIDE
        - 51.3% of the users' group sense and activity are symmetrical with the concepts from the SIDE model.
        - Only 3% of the user's behavior is not parallel with SIDE in our platform.
        - The rest of the users are out of scope as they have moderate, not strong/weak group sense among them.

8.3 Related Work

8.3.1 Different Argumentation Platform

Many researchers have worked on analyzing user opinion and behavior in cyber argumentation systems. Some platforms, like opinion space [11] and Considerit [12], focused on analyzing user's opinions and how interacting with different users impacts their overall opinion on different social and political issues. Other platforms such as Citizen report card [13], Open Town Hall [14], and California report card [15] focused on surveying collective user opinion on important issues from a public service perspective. In these systems, users do not have many ways of interacting with others, so there is less opportunity to actively exchange perspectives and ideas [16], which limits their use to analyze different user interaction and opinion dynamics related phenomena. In this work, we use our interactive ICAS platform to analyze user behavior and opinion and different group related phenomena.

8.3.2 Use of Different Social Science Theories in Argumentation and Social Media Platform

Many social science and psychological theories which are based on real-life empirical studies have been used extensively in different user behavior and interactions related models in web and social media platform. [17] used the theory of planned behavior with additional variables of self-identity and belongingness, to predict the high-level use of social networking among a sample of young people. [18] used the key ideas from social cognitive theory to empirically analyze the interactions among individuals in a web-based self-regulated learning application. [19] also applied the social cognitive theory for behavior modeling to train users for computer skills. This theory is also used to analyze and understand why people use social networking sites [20]. [21] used the social capital theory to analyze how the contents are used for collaboration from Nigerian University web sites.

[22] applied the behavior-change theories to analyze the behaviors that give rise to violence and injury and used in injury prevention methods.

Social and psychological theories have also been used to design and model user interfaces. [23] used the Diffusion of innovation theory in interface design that supports the twitter hashtag use and access for hashtag management and other information for decision making. [24] analyzed different cognitive psychological perspectives on social learning theories and how they can be used in the design and implementation of online learning. These researches show that although these theories are based on real-life empirical studies, they can be used in an online environment in different social, psychological phenomena analysis, user behavior modeling, and user interface design.

8.3.3 Use of Social Identity Theory and Social Identity Model of Deindividuation Effects in different web applications

Social Identity Theory [8] and Social Identity Model of Deindividuation Effects [9] have been used to analyze the concept of self, and the effect of anonymity on different web and social media applications. Different concepts from these two theories have been used to analyze the effect of anonymity and group norms on aggressive language [4], group polarization [3], self-awareness and argument quality [5], the impact of uniform virtual appearance on the individual inclination to conform to majority opinion [6], and the effect of transformed self-presentation on user behavior [7] in online platforms. There are many other examples where the self-concepts from these two theories have been used to analyze and model user behavior and the effect of anonymity. To our knowledge, there is no prior work that tried to examine the validity of these theories in their web platform before applying them in different social and psychological phenomena analysis like our work did present in this paper.

8.4 ICAS System

We developed an Intelligent Cyber Argumentation System (ICAS), where participants can join and engage with each other to discuss different issues. ICAS can automatically determine the user's opinion from the discussion. ICAS platform is the enhanced version of the argumentation system developed for previous work [25]. For more details of the ICAS system and it's architecture, please refer to section 4.1.

8.4.1 Mining User Opinion

In our system, the user's opinion value on a position is automatically calculated using the agreement values from all the arguments users posted under that corresponding position. This reconciliation process is handled by ICAS's built-in argument reduction system [26], which uses artificial intelligence, fuzzy logic, and linguistic rules. For more details on the argument reduction method, please refer to [26, 27].

8.5 Group Identity and Group Influence in the User Activity Concept

In this section, we discussed the "Group identity" and "Group Influence in User Activity" concept according to Social Identity Theory (SIT) and Social Identity Model of Deindividuation Effects (SIDE) and the key concepts we will be examining from these theories. Then we discussed how we perceived the "Group identity" and "Group Influence in User Activity" concept in the perspective of our ICAS platform.

8.5.1 Group Identity and Group Influence Concept according to SIT and SIDE

In this subsection, we discussed briefly Social Identity Theory (SIT) and Social Identity Model of Deindividuation Effects (SIDE) to introduce the concept of "Group Identity" and "Group Influence on User Activity."

1) Social Identity Theory (SIT)

According to social identity theory (SIT) [8], people psychologically categorize and identify themselves as part of existing social groups. This is a person's psychological concept or sense that they belong to a certain group. This group membership gives individuals a sense of social identity and belonging to the world. This kind of categorization derives from the normal cognitive process or the tendency to group things together. Even without any real-life interaction with others, this psychological phenomena (social group categorization and identification with a social group) influences people's attitude and behavior to other people who they think are in their group and not in their group.

2) Social Identity Model of Deindividuation Effects (SIDE)

SIDE [9] explains the effects of anonymity in user behavior in a group interaction environment. Anonymity refres to the situation where users cannot visually identify each other with their real name, face, or ID. In a group environment where people interact with each other anonymously, visual anonymity hides individual features and interpersonal differences. Individuals have decreased visibility as a separate individual, and it depersonalizes social perception of self and others. Thus anonymity enhances the loss of self-awareness and increases group identity. Group identity is an individual's concept or sense of belonging to the group or how much an individual identifies as a member of the group. According to SIDE, if there is some basis of sharedness among individuals to perceive themselves and others as members of one group, then anonymity will enhance the prominence of group identity among individuals in a group environment. People will tend to perceive self and others in terms of stereotypic group features and will be influenced by the group accordingly. The more an individual identifies with a group, the more aligned his/her behaviors will be with the group. If the group sense is not more prominent than the individual

identity, then the individuals will act on their own; it will not be consistent with the group. This behavior includes the attitude and kind of activities towards the members of the same group and other groups.

8.5.2 Key Concepts of the SIDE model for Examination

We will design an experiment to validate the following three key concepts from SIDE in our platform:

- Concept 1: Anonymity increases the importance of group identity among individuals in a group interaction environment.
- Concept 2: Individuals with strong group identity will be influenced to be more similar to their group in their activity.
- Concept 3: Weak group identity does not influence user behavior, so user behavior will be more dissimilar with their group.

To analyze these concepts, we need to quantify the idea of group sense among individuals and individual and group activity in the perspective of our ICAS platform.

8.5.3 Group Identity and Group Influence Concept in Perspective of ICAS Platform

Previous research [10] has found that similar opinion in the social or political issue also drives the sense of groupness which Social Identity Theory refers to among individuals. In our system, users discuss different social and political issues with a different spectrum of similar/dissimilar opinionated users. Even though users do not explicitly join or categorize themselves with any particular opinionated group, according to social identity theory [8] and [10], users in our system should have psychological group sense with similar opinionated users to trigger a group identity among them. In the user behavior perspective, users support or attack other users' arguments in

the discussion of different issues in our platform. Now the group sense might influence their support/attack behavior to the members of the same group and the members of other groups. Individuals might get influenced by their group members in their support-attack pattern to other in-group and out-group members. We will use this support/attack pattern to ingroup and outgroup members of each individual as an individual's activity and the average support attack pattern of the group members as a group activity in our experiment.

8.6 Design of Experiment to validate the Key Concepts

In this section, we discussed the experimental design in order to examine the above mentioned key concepts of SIDE in our platform. We first gave a brief overview of the whole experimental steps, description of the steps, and finally discussed the statistical significance test to confirm whether our observation is random or not.

8.6.1 Overview of the Whole Experiment

In our experiment, we will go through the following steps to examine the key concepts' validity from SIDE:

- We will divide the whole user-space into different opinionated groups.
- For each user in each group,
    - We will measure the user's group identity value.
    - Based on the group identity value, we will label the user as a user with a strong/medium/weak group identity.
    - We will measure the user's activity similarity value with the group.
    - Based on the activity similarity value, we will label the user as a user with strong/medium/weak similar activity with the group.

- We will put the user in one of the nine predefined categories based on the strong/medium/weak labels of “Group Identity” and “Similar Activity with the Group.” Each category represents a particular group identity label, and a similar activity label, such as (Strong Group Identity & Strong Similar Activity with the Group) is a category.

- We will analyze the distribution of users in these nine categories in order to examine the validity of the concepts from the SIDE.
- We will perform a statistical significance test based on the distribution of users in the nine categories to make sure that our observation is not random.

8.6.2 Experimental Details

In this section, we discussed our experimental steps briefly, how we quantified different concepts, and performed analysis on user distribution to examine the concepts of SIDE.

1) Deriving Opinionated User Groups

As a first step, we need to group users based on their opinion in an issue using a clustering algorithm. In our platform, users did not participate in all the positions of an issue. So, the collected user-opinion dataset contains lots of missing values. Grouping with missing values introduces error and bias in group analytics[28]. We used the CSCCF opinion prediction model [29] to predict the missing opinion values. For more details on the group identification process, please refer to section 6.2.

2) Measure Group Identity and Identify Users with Strong/ Medium/ Weak Group Identities

In this step, we used the generated user groups from the previous step and measured the group identity value for each user and labeled them as users with strong/medium/weak group identities.

Group identity refers to how much an individual identifies with the group. We will use the difference between individual user opinion and average group opinion to define how much a user identifies with the group. The rationale for this is that a more similar opinionated user with the group will have a higher probability of finding users like him in the group to trigger the sharedness or group sense than a less similar opinionated user. This group identity value is calculated using the following equation:

$$Group\ Identity = 1 - \frac{\sum_{i=0}^{n} abs(u_i - p_i)}{K}$$

Here, u_i is the user agreement value, and p_i is group mean agreement value at position i. n is the number of positions in an issue. K is the maximum opinion value difference. Based on the group identity value for each user, we marked them as users with strong/medium/weak group identities using threshold values. If the group identity value is greater than 70 percent, we labeled that user as a user with a strong group identity, if it is lower than 30 percent than as user with weak group identity, otherwise as a user with medium group identity.

3) Measure User Activity Similarity with the Group and Identify users with Strong/ Medium/ Weak Activity Similarity with the Group

In this step, we measured the activity similarity value for the users in the generated user groups from the first step and labeled them as users with strong/medium/weak similar activity with the group. In our ICAS platform, users support and attack other users in different position discussions. We used this support-attack pattern as an individual activity and group activity.

For each user, we will generate a support-attack vector (SAV) from all the supports and attacks towards the members within the user's own group and out-groups in all the positions of an issue. If there are n groups in an issue, there will be a support value and attack value for each group in

the SAV. Let, user x supported m times and attacked n times in total to all the users in the position discussions. If user x supported p times and attacked q times to the members who belong to group a, then the support value to Group a, G_a^S and the attack value to Group a, G_a^A for user x can be formalized in the following way

$G_a^S = p/m$

$G_a^A = q/n$

If there are n user groups in all the position discussions in an issue, then the support-attack vector for user x, can be formalized in the following way

$SAV_x = [G_1^S, G_1^A, G_2^S, G_2^A, \ldots\ldots\ldots\ldots, G_n^S, G_n^A]$

For group-level activity, we aggregated the support and attack values from all the users who belong to the group. Let, all members of group i in total supported s times and attacked t times other users. If all the members of group i support u times and attack v times to all the members of group j, then the support-attack values for group i to group j can be formalized as

$G_{ij}^S = u/s$

$G_{ij}^A = v/t$

In there are n groups in total, then the support–attack vector for group i can be formalized as follows:

$SAV_i = [G_{i1}^S, G_{i1}^A, G_{i2}^S, G_{i2}^A, \ldots\ldots\ldots\ldots, G_{in}^S, G_{in}^A]$

If the user x belongs to group i, we can measure the activity similarity between user x and group i using the cosine similarity between SAV_x and SAV_i.

Similarity of activity (user x, group i) = Cosine Similarity (SAV_x , SAV_i)

In this way, we can measure the activity similarity between a user and the group user belongs to. If a user's activity is similar to the group, then the support attack pattern to the in-group, out-group, should be similar to the average support attack pattern of the group.

4) User Distribution Analysis

In this step, we will analyze the user distribution based on their group identity and similar activity with the group labels. We have strong/medium/weak labels for group identity and strong/medium/weak labels for similar activity with the group. With these labels, we defined nine categories where each category represents a particular combination of these two variables, such as "Strong Group Identity" & "Strong similar activity with the group." According to the SIDE model, users with strong group identity should have similar strong activity with the group, and users with weak group identity should have similar weak activity with the group. Users with medium group identity are out of scope for the SIDE model. This category information and symmetry with the SIDE model are summarized in the following table.

Table 8.1. Group Identity and Group Activity Similarity Categorization

	Strong Group Activity Similarity	Medium Group Activity Similarity	Weak Group Activity Similarity
Strong Group Identity	Category 1: Consistent with SIDE	Category 2: Out of scope	Category 3: Not Consistent with SIDE
Medium Group Identity	Category 4: Out of scope	Category 5: Out of scope	Category 6: Out of scope
Weak Group Identity	Category 7: Not Consistent with SIDE	Category 8: Out of scope	Category 9: Consistent with SIDE

For each group, we put every user in a category based on their group identity and activity similarity label and measured the percentage of users in each category. With this user distribution in different categories, we will make the following observations:

- To examine the validity of concept 1, we will be looking at the percentage of users with strong group sense in each group. If the majority or a significant percentage of the users have strong group identity, then concept 1 from the SIDE model is valid in our platform.
- To examine the validity of concept 2 and concept 3, we will measure the percentage of users falls in category 1 (Strong Group Identity & Strong Similar Activity with Group) and 9 (Weak Group Identity & Weak Similar Activity with Group) as their behaviors are consistent with the SIDE model. This observation will also provide us the percentage of users for which the SIDE model is valid in our platform
- To measure the percentage of user's behavior is not consistent with the SIDE model, we will observe the percentage of users falls in category 3 and 7, as their behavior is not consistent with the SIDE model.
- To measure the percentage of users' behavior is out of scope in our platform, we will observe the percentage of users falls under category 2, 4, 5, 6, and 8. They either have a medium group identity or similar medium activity with the group. Their behavior is not explained by the SIDE model.

5) Statistical Significance Test:

In this step, we performed a statistical significance test for each group in all issues to confirm the significant relationship we observed between "Group Identity" and "Activity Similarity with the Group" is not random. Chi-Square test is a popular approach to test the association between two variables. However, the assumption of the Chi-square test is that the expected value in each cell

needs to be greater than 5, which was not the scenario in many of our cases. So we used Fisher's exact test [30] to analyze the association between "Group Identity" and "Similar Activity with the Group" Variables. Fisher's exact test will give us the P-value. The P-value represents the probability that the "Group Identity" and "Activity Similarity with the Group" are independent. It also represents the probability to reject the Null Hypothesis. From the average P-value, we can validate where the observed association between these two variables is random or not.

8.7 Experiment

8.7.1 Empirical Data Description

We organized an empirical study in an entry-level sociology class in the spring of 2018 session and collected dataset for this work.

8.7.2 Experimental Result

In this section, we discussed the experimental results for each group in each issue. This result contains the social profile information of the group, political leanings, percentage of users falls under different combinations of (Group Identity, Similar Activity with the Group) variables. We reported the percentage of users falls under Cat1 (Strong, Strong), Cat3 (Strong, Weak), Cat7 (Weak, Strong), and Cat9 (Weak, Weak) values of (Group Identity (GI), Similar Activity with the Group (SA)) variables. Users in Cat1 and Cat9 are consistent with SIDE, and users in Cat3 and Cat7 are not consistent with SIDE. The rest of the users fall under not in the scope of SIDE as they either have medium group identity or medium activity similarity value with the group. For more details on the political leanings and social profile information of the groups, please refer to section 6.2.

1) Experimental Results for "Guns on Campus" Issue

In the guns on campus issue, participants were clustered into five groups (G0, G1, G2, G3, and G4). Based on their support and attack values towards the conservative and liberal positions, G1 is the most conservative group; they agree that concealed carry permit is enough to carry a gun on campus. They disagree on any restriction to carry firearms on the campus. G3 is the most liberal group; they strongly support that college campuses should not allow students to carry firearms under any circumstances. G4, the overall in the middle group supports mostly that carry permit is not enough; some restrictions should be applied to allow students to carry firearms on campus. G2, the overall conservative group, is more approving for students carrying guns but with restrictions like special permission, additional training, or test than banning guns on campus totally or giving students full freedom to carry guns on campus. G0, the overall liberal group, supports but not as strongly as G3, that college campuses should not allow students to carry firearms under any circumstances.

Table 8.2: Group Identity and Activity of Groups for Guns on Campus Issue.

Group	Strong Group Identity (%)	Weak Group Identity (%)	Cat#1 (Strong GI, Strong SA) (%)	Cat#3 (Strong GI, Weak SA) (%)	Cat#7 (Weak GI, Strong SA) (%)	Cat#9 (Weak GI, Weak SA) (%)
G0	72	0	51	5	0	0
G1	51	0	37	0	0	0
G2	73	0	53	2	0	0
G3	55	2	38	0	2	0
G4	83	0	56	3	0	0

All the groups have a significant portion of their users with strong group sense. G4 has the highest percentage of users with strong group identity, and G1 has the lowest percentage of users with

strong group sense. In all groups we can see the strong group influence in individual behavior, a significant portion of the users falls under the strong group identity and similar strong activity with the group (cat 1) and a marginal portion in strong group identity but similar weak activity with the group (cat3). 56% of G4 users are in cat1, and only 3% of users of G4 are in cat3. There is no users in weak group identity and weak similar activity with the category (cat9) in all the groups. G3 only contains 2% of its users in cat7; other groups do not include any users in this category. In this issue, users are showing group influence in their behavior (cat1 and cat9) as per the SIDE model, only a very little percentage of users are in cat3 and cat7, their behavior is not explained by the SIDE model.

2) Summarized Experimental Result for Other Three Issues

In this section, we discussed the summarized social profile information and group activity experimental results for the other three issues in our platform.

a) Summarized Group Identity and Activity Experimental Results

For each issue, the number of groups and the average percentage of users with strong and weak group identity, and in different categories among the groups is presented in table 8.3.

Table 8.3: Group Identity and Activity of Groups for Other Issue.

Issue Name	Number of Group	Strong Group Identity (%)	Weak Group Identity (%)	Cat#1 (Strong GI, Strong SA) (%)	Cat#3 (Strong GI, Weak SA) (%)	Cat#7 (Weak GI, Strong SA) (%)	Cat#9 (Weak GI, Weak SA) (%)
Religion & Medicine	6	79	3	47	5	0	0
Same Sex Couples & Adoption	4	68	3	51	6	3	0
Government & Healthcare	5	77	0	57	3	0	0

From all these experimental results, we can see that

- Anonymity and similar Opinion is triggering a strong group sense among the majority of the user (73.5% on average) in our platform.
- Strong Group Sense is influencing users to be more similar in their activity with the group they belong to in the majority of the users.
    - On average, 69.9% of the users with strong group sense also have strong similar activity with the group.
    - Only a marginal percentage (on average, 3.7%) of users have weak similar activity even though they have strong group sense among them.
- On average, 51.3% of users' behavior is symmetrical with the concept of SIDE. They have either (Strong Group Sense & Strong Similar Activity with the Group) or (Weak Group Sense & Weak Similar Activity with the Group).
- On average, 3% of users' behavior is not symmetrical with the concept of SIDE. They have either (Strong Group Sense & Weak Similar Activity with the Group) or (Weak Group Sense & Strong Similar Activity with the Group).
- The rest of the users are out of scope for the SIDE model, as they do not have strong or weak group sense among them.

8.7.3 Statistical Significance Test:

We performed Fisher's exact test [30] to make sure that our observation on the "Group Identity" and "Activity Similarity with the Group" is not random. The P-value represents the probability of rejecting the Null Hypothesis or that these two variables are independent. For each group, the p-value is presented in table 8.4.

The average p-value for each issue across groups is presented in the following table. From the results, we can see that on average, there is a 14.13% probability that the "Group Identity" and "Activity Similarity with the Group" variables are independent. This statement can also be said as there is an 85.87% probability that these variables are associated with each other. From the statistical point of view, there is not a highly significant relationship between these two variables, but there is a near statistical significance or moderately significant relationship between these two variables.

Table 8.4: P-value of Fisher's Exact Test in Different Issues

Issue Name	P-Value (Average)
Gun Issue	0.1304
Religion & Medicine Issue	0.1755
Samesex and Adoption Issue	0.1277
Government and Healthcare Issue	0.1314

From our observation and statistical significance test, we can see that even though without any explicit group joining, anonymity and psychological group identification is triggering users' similar activity in the issue discussions in our platform. Users are supporting and attacking the other individuals within their own group and outside their group in a similar pattern even without any explicit or coordinated decision among themselves.

8.8 Conclusion

Analyzing the impact of different types of user interactions on individual user behavior is one of the key requirements for user behavior modeling and different social and psychological phenomena analysis in cyber argumentation platforms. The key ideas and different aspects of many social science theories have been used in many web applications. As these experiments are done in real life, face to face communication, they might not be valid in computer-mediated

communication. In this paper, we modeled an experiment to quantify and examine whether the key concepts of the SIDE is valid using the concept from SIT. Our results show that 73.5% of our total users have strong group sense, and 51.3% of the user's behavior is symmetrical with SIDE in our platform. Only a marginal portion (3%) of our users' behavior is not symmetrical as per SIDE; other users are out of score for SIDE. These results show that anonymity and the concept of group sense are impacting a significant portion of users' behavior in our platform. This work opens the door of using different concepts from SIT, SIDE into our discourse platform. These theories can enhance user opinion analysis and be used to neutralize the problematic situations such as group polarization [3] in our argumentation platform.

8.9 References

[1] Mark Klein. 2015. The CATALYST Deliberation Analytics Server. Social Science Research Network, Rochester, NY. Retrieved August 19, 2019 from https://papers.ssrn.com/abstract=2962524

[2] Joseph Sirrianni, Xiaoqing Liu, and Douglas Adams. 2018. Quantitative Modeling of Polarization in Online Intelligent Argumentation and Deliberation for Capturing Collective Intelligence. In 2018 IEEE International Conference on Cognitive Computing (ICCC), 57–64. DOI:https://doi.org/10.1109/ICCC.2018.00015

[3] David G. Myers and Helmut Lamm. 1975. The Polarizing Effect of Group Discussion: The discovery that discussion tends to enhance the average prediscussion tendency has stimulated new insights about the nature of group influence. American Scientist 63, 3 (1975), 297–303.

[4] Leonie Rösner and Nicole C. Krämer. 2016. Verbal Venting in the Social Web: Effects of Anonymity and Group Norms on Aggressive Language Use in Online Comments: Social Media + Society (August 2016). DOI:https://doi.org/10.1177/2056305116664220

[5] Eun-Ju Lee. 2007. Deindividuation Effects on Group Polarization in Computer-Mediated Communication: The Role of Group Identification, Public-Self-Awareness, and Perceived Argument Quality. Journal of Communication 57, 2 (2007), 385–403. DOI:https://doi.org/10.1111/j.1460-2466.2007.00348.x

[6] Junghyun Kim and Hee Sun Park. 2011. The effect of uniform virtual appearance on conformity intention: Social identity model of deindividuation effects and optimal distinctiveness theory. Computers in Human Behavior 27, 3 (May 2011), 1223–1230. DOI:https://doi.org/10.1016/j.chb.2011.01.002

[7] Nick Yee and Jeremy Bailenson. 2007. The Proteus Effect: The Effect of Transformed Self-Representation on Behavior. Hum Commun Res 33, 3 (July 2007), 271–290. DOI:https://doi.org/10.1111/j.1468-2958.2007.00299.x

[8] Henri Tajfel and John C. Turner. 2004. The Social Identity Theory of Intergroup Behavior. In Political Psychology (0 ed.), John T. Jost and Jim Sidanius (eds.). Psychology Press, 276–293. DOI:https://doi.org/10.4324/9780203505984-16

[9] S. D. Reicher, R. Spears, and T. Postmes. 1995. A Social Identity Model of Deindividuation Phenomena. European Review of Social Psychology 6, 1 (January 1995), 161–198. DOI:https://doi.org/10.1080/14792779443000049

[10] Ana-Maria Bliuc, Craig McGarty, Katherine Reynolds, and Daniela Muntele. 2007. Opinion-based group membership as a predictor of commitment to political action. European Journal of Social Psychology 37, 1 (2007), 19–32. DOI:https://doi.org/10.1002/ejsp.334

[11] Siamak Faridani, Ephrat Bitton, Kimiko Ryokai, and Ken Goldberg. 2010. Opinion Space: A Scalable Tool for Browsing Online Comments. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10), ACM, New York, NY, USA, 1175–1184. DOI:https://doi.org/10.1145/1753326.1753502

[12] Travis Kriplean, Jonathan Morgan, Deen Freelon, Alan Borning, and Lance Bennett. 2012. Supporting Reflective Public Thought with Considerit. In Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work (CSCW '12), ACM, New York, NY, USA, 265–274. DOI:https://doi.org/10.1145/2145204.2145249

[13] Participation & Civic Engagement - Citizen Report Card and Community Score Card. Retrived August 20, 2019 from http://web.worldbank.org/WBSITE/EXTERNAL/TOPICS/EXTSOCIALDEVELOPMENT/EXTPCENG/0,,contentMDK:20507680~pagePK:148956~piPK:216618~theSitePK:410306,00.html

[14] The Extent of Public Participation. Retrieved January, 26 2020 from https://icma.org/articles/pm-magazine/extent-public-participation

[15] Civic Media Project: The California Report Card Version 1.0. Retrieved January 8, 2019 from http://civicmediaproject.org/works/civic-media-project/thecaliforniareportcard

[16] Matti Nelimarkka, Brandie Nonnecke, Sanjay Krishnan, Tanja Aitumurto, Daniel Catterson, Camille Crittenden, Chris Garland, Conrad Gregory, Ching-Chang (Allen) Huang, Gavin Newsom, Jay Patel, John Scott, and Ken Goldberg. 2014. Comparing Three Online Civic Engagement Platforms using the Spectrum of Public Participation. In THE INTERNET, POLICY & POLITICS CONFERENCES, Oxford. Retrieved January 8, 2019 from https://escholarship.org/uc/item/0bz755bj

[17] Emma Pelling and Katherine M. White. 2009. The theory of planned behaviour applied to young people's use of social networking websites. Cyberpsychology & Behavior 12, (2009), 755–759.

[18] Shu-Ling Wang and Sunny S. J. Lin. 2007. The application of social cognitive theory to web-based learning through NetPorts. British Journal of Educational Technology 38, 4 (2007), 600–612. DOI:https://doi.org/10.1111/j.1467-8535.2006.00645.x

[19] Deborah R. Compeau and Christopher A. Higgins. 1995. Application of Social Cognitive Theory to Training for Computer Skills. Information Systems Research 6, 2 (1995), 118–143.

[20] Kuan-Yu Lin and Hsi-Peng Lu. 2011. Why people use social networking sites: An empirical study integrating network externalities and motivation theory. Computers in Human Behavior 27, 3 (May 2011), 1152–1161. DOI:https://doi.org/10.1016/j.chb.2010.12.009

[21] Samuel C. Utulu and Maryknoll A. Okoye. 2010. Application of social capital theory to Nigerian university web sites. The Electronic Library (February 2010). DOI:https://doi.org/10.1108/02640471011023450

[22] Andrea Carlson Gielen and David Sleet. 2003. Application of Behavior-Change Theories and Methods to Injury Prevention. Epidemiol Rev 25, 1 (August 2003), 65–76. DOI:https://doi.org/10.1093/epirev/mxg004

[23] Hsia-Ching Chang. 2010. A new perspective on Twitter hashtag use: Diffusion of innovation theory. Proceedings of the American Society for Information Science and Technology 47, 1 (2010), 1–4. DOI:https://doi.org/10.1002/meet.14504701295

[24] Janette R. Hill, Liyan Song, and Richard E. West. 2009. Social Learning Theory and Web-Based Learning Environments: A Review of Research and Discussion of Implications. American Journal of Distance Education 23, 2 (2009), 88–103.

[25] X. Liu, E. Khudkhudia, L. Wen and V. Sajja, An Intelligent Computational Argumentation System for Supporting Collaborative Software Development Decision Making, 2010, pp. 167–180. doi:10.4018/978-1-60566-758-4.ch009

[26] Scott Sigman and Xiaoqing Frank Liu. 2003. A computational argumentation methodology for capturing and analyzing design rationale arising from multiple perspectives. Information & Software Technology 45, (2003), 113–122. DOI:https://doi.org/10.1016/S0950-5849(02)00187-8

[27] Xiaoqing (Frank) Liu, Eric Christopher Barnes, and Juha Erik Savolainen. 2012. Conflict Detection and Resolution for Product Line Design in a Collaborative Decision Making Environment. In Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work (CSCW '12), ACM, New York, NY, USA, 1327–1336. DOI:https://doi.org/10.1145/2145204.2145402

[28] Shichao Zhang, Jilian Zhang, Xiaofeng Zhu, Yongsong Qin, and Chengqi Zhang. 2008. Missing Value Imputation Based on Data Clustering. In Transactions on Computational Science I, Marina L. Gavrilova and C. J. Kenneth Tan (eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 128–138. DOI:https://doi.org/10.1007/978-3-540-79299-4_7

[29] Md Mahfuzer Rahman, Joseph W. Sirrianni, Xiaoqing (Frank) Liu, and Douglas Adams. 2019. Predicting opinions across multiple issues in large scale cyber argumentation using collaborative filtering and viewpoint correlation. In The Ninth International Conference on Social Media Technologies, Communication, and Informatics (SOTICS 2019).

[30] R. A. Fisher. 1922. On the Interpretation of $\chi 2$ from Contingency Tables, and the Calculation of P. Journal of the Royal Statistical Society 85, 1 (1922), 87–94. DOI:https://doi.org/10.2307/2340521

Chapter 9: Conclusion

The developed ICNC-MF, Ex-ICNC-RME web API recommendation models for mashup applications, and the CSCCF opinion prediction model for the cyber-argumentation platform presented in this work enhanced the accuracy and quality of prediction and recommendation results for online discussion platforms and mashup software applications. These algorithms and models integrate identified improvement scopes with the current best-performing solutions efficiently, utilizing available information from multiple sources such as content, network, or user interactions in an optimized fashion. Also, they address data sparsity, cold start, etc., research challenges that cause traditional prediction and recommendation approaches to fail. Besides, the Group Representation and Group Identity metrics, Intra-group interaction, and Inter-group interaction models incorporate social psychological theories with graph mining and computational methods; they can ensure a better understanding of user behavior and online discussions concerning the representation and interaction of different user groups in the cyber-argumentation platform.

Appendix

Appendix A: IRB Protocol Approval Letter

UNIVERSITY OF ARKANSAS

To:	Xiaoqing Liu JBHT 0504
From:	Douglas James Adams, Chair IRB Committee
Date:	03/17/2020
Action:	**Expedited Approval**
Action Date:	03/17/2020
Protocol #:	1710077940R002
Study Title:	Empirical Study of Structured Cyber Argumentation
Expiration Date:	11/12/2020
Last Approval Date:	03/17/2020

The above-referenced protocol has been approved following expedited review by the IRB Committee that oversees research with human subjects.

If the research involves collaboration with another institution then the research cannot commence until the Committee receives written notification of approval from the collaborating institution's IRB.

It is the Principal Investigator's responsibility to obtain review and continued approval before the expiration date.

Protocols are approved for a maximum period of one year. You may not continue any research activity beyond the expiration date without Committee approval. Please submit continuation requests early enough to allow sufficient time for review. Failure to receive approval for continuation before the expiration date will result in the automatic suspension of the approval of this protocol. Information collected following suspension is unapproved research and cannot be reported or published as research data. If you do not wish continued approval, please notify the Committee of the study closure.

Adverse Events: Any serious or unexpected adverse event must be reported to the IRB Committee within 48 hours. All other adverse events should be reported within 10 working days.

Amendments: If you wish to change any aspect of this study, such as the procedures, the consent forms, study personnel, or number of participants, please submit an amendment to the IRB. All changes must be approved by the IRB Committee before they can be initiated.

You must maintain a research file for at least 3 years after completion of the study. This file should include all correspondence with the IRB Committee, original signed consent forms, and study data.

cc: Douglas James Adams, Investigator
Joseph W Sirrianni, Investigator
Md Mahfuzer Rahman, Investigator
Najla Althuniyan, Investigator
Zhang Hu, Investigator

Page 1 of 1