Research Article

DETECTING FAKE PROFILES IN ONLINE WITH CLUSTERING TECHNIQUES IN SOCIAL MEDIA STREAMS

Dr. M. Prabakaran,
Assistant Professor, PG & Research Department of Computer Science
Government Arts College (Autonomous), Karur-639005.
Email: captainprabakaran@gmail.com

ABSTRACT

Leveraging clustering techniques, the study aims to enhance the identification of fraudulent activities. The proposed approach addresses the challenges of security and efficiency, offering a robust solution for tracking and mitigating the impact of fake profiles on social media. This research investigates the usability and challenges associated with Wiki blog content management systems, emphasizing natural language interfaces and sentiment analysis. Employing a Systematic Mapping Study, the study identifies key trends, challenges, and applied Human-Computer Interaction techniques in Wiki blog content management usability. It introduces Xatkit, a model-driven development framework, to address challenges and provide a higher level of abstraction in Wiki blog content management definition. The research explores sentiment analysis, proposing cross-domain sentiment classification, sentiment-sensitive thesaurus creation, and the optimization of sentiment classification using genetic algorithms. Results indicate the effectiveness of deep learning-based systems in handling emotional requests on social media. The study concludes with an overview of primary studies, highlighting the growing interest in Wiki blog content management usability. Despite advancements, the research underscores the need for further exploration and universally applicable guidelines in Wiki blog content management usability.

KEYWORDS

Online Fake Profiles, Social Media Streams, Clustering Techniques, Fraudulent Activities, Security, Efficiency, Profile Tracking, Social Media Platforms, Cybersecurity.

INTRODUCTION

Software vendors and service providers are increasingly delivering services to users through the Internet. Largescale web services, such as maps, search, and social networking, have proliferated, attracting hundreds of millions of users worldwide [1, 2]. On the client side, these services heavily leverage Asynchronous Javascript and XML to provide a seamless and consistent user experience across devices and form factors. Behind the scenes, significant amounts of data and computation are provided by servers in the cloud. Many web services are extremely complex, since they aim to match or even exceed the rich user experience offered by traditional desktop application. For instance, the "driving directions" webpage of Yahoo Maps comprises about 110 embedded objects and 670KB of Javascript code. These objects are retrieved from many different servers, sometimes even from multiple data centers and content distribution networks.

These dispersed objects meet only at a client machine, where they are assembled by a browser to form a complete webpage. Since service providers lack object-level measurements obtained from clients, it is hard for them to assess and study user-perceived performance. Moreover, there exist a plethora of dependencies between different objects. Many objects cannot be downloaded until some other objects are available [3]. For instance, an image download may have to await a Javascript download because the former is requested by the latter. These multiple factors make it highly challenging to understand and predict the performance of web services. The performance of web services has direct impact on user satisfaction. Poor page load times (PLT) result in low service usage, which in turn may undermine service income. For instance, a study by Amazon reported roughly 1% sales loss as the cost of a 100 ms extra delay [4-7].

Another study by Google found a 500 ms extra delay in display search results may reduce revenues by up to 20% [16]. Even worse, users may simply abandon a service provider for another offering, as switching barriers are often low. Ideally, service providers would like to predict the effects of potential optimizations before actual deployment. Yet it is seldom clear what benefits various possible options for improvement might bring a service provider – whether optimization to the object structure of the page, or optimizations in the manner in which content is placed and delivered over the Internet [8].

Platforms for instant messaging have become one of the most popular ways to communicate and exchange information. The majority of them offer built-in assistance for integrating Wiki blog content

management applications, which are automated conversational agents equipped to communicate with platform users [9]. Wiki blog content management has been effective for automating processes and enhancing user experience in a variety of settings, including automated customer support, education, and e-commerce. However, even though numerous systems have lately appeared for managing the content of Wiki blogs, building and deploying them still requires significant technical expertise [10].

Wiki blog content management are also increasingly used to facilitate software engineering activities like automating deployment tasks, assigning software bugs and issues, repairing build failures, scheduling tasks like sending reminders, integrating communication channels, or for customer support. In this context, we explored the use of Wiki blog content management for domain modelling in previous work. Modelling Wiki blog content management can be embedded within social networks to support collaboration between different stakeholders in a natural way and enable the active participation of non-technical stakeholders in model creation [13].

The necessity to swiftly develop complicated Wiki blog content management supporting NL processing, customized knowledge repository definition, and intricate action responses incorporating external service composition has been highlighted by the high demand for and interest in Wiki blog content management applications. A thorough comprehension of the API of the desired instant messaging networks as well as third-party services that must be integrated are all necessary for the construction of Wiki blog content management, making it a difficult task [14]. Intents are defined via training phrases. These phrases may include parameters of a certain type (e.g., numbers, days of the week, countries). The parameter types are called entities. Most platforms come with predefined sets of entities and permit defining new ones.

Some platforms permit structuring the conversation as an expected flow of intents. For this purpose, a common mechanism is providing intents with a context that stores information gathered from phrase parameters, and whose values are required to trigger the intent. In addition, there is normally the possibility to have a fallback intent, to be used when the bot does not understand the user input [15].

Natural Language Processing (NLP), pattern matching algorithms, and parsing are frequently used in wiki blog content management design to represent wiki blog content management knowledge. The latter approach has gained popularity because it uses Machine Learning (ML) techniques to comprehend user input and offer user-friendly interfaces to design conversational flow. Examples of these services include Dialog Flow and IBM Watson Assistant. Pereira and Dáz, however, have recently reported that Wiki blog content management applications cannot be reduced to simple language processing capabilities and must also take into account other factors like complex system engineering, service integration, and testing when designing them. As a matter of fact, the conversational aspect of the application is typically the front-end of a bigger system that comprises data storage and application execution as part of the Wiki blog content management response to the user intent. As a result, we developed an application to manage the content of a Wiki blog that incorporates a recognition engine to identify user intentions and an execution component that handles complicated event processing and is represented as a series of actions.

The Wiki blog content management designer has to provide a domain meta-model and optionally, a NL configuration model. The latter is used to optionally annotate classes, attributes and features with synonyms, and the source of references with a name to refer to its backward navigation. From this information, we generate the Wiki blog content management intents and entities.

The generated Wiki blog content management also contains actions, required to perform the query on a modeling backend, which we call the execution model. This execution model contains a set of execution rules that bind user intentions to response actions as part of the Wiki blog content management behavior definition.

For each intent in the Intent model, we generate the corresponding execution rule in the execution model using an event-based language that receives as input the recognized intent together with the set of parameter values matched by the NL engine during the analysis and classification of the user utterance.

All the execution rules follow the same process: the matched intent and the parameters are used to build an OCL-like query to collect the set of objects the user wants to retrieve. The intent determines the type of query to perform (e.g., all Instances, select, etc.), while the parameters identify the query parameters, predicates, and their composition. The query computation is delegated to the underlying modeling platform (see next section), and the returned model elements are processed to build a human-readable message that is finally posted to the user by the bot engine.

By defining Wiki blog content management at a higher level of abstraction, this effort tries to address all of these problems. In order to achieve this, we present Xatkit, a cutting-edge model-based Wiki blog content management development framework that tries to answer this question utilizing Model Driven Engineering (MDE) techniques, including platform independent bot definitions, runtime interpretation, and domain specific languages. In fact, Xatkit incorporates a specialized modeling language for Wiki blog content management to express user intentions, computational activities, and callable services, fusing them together in rich conversation flows.

Conversations can either be started by a user awakening Xatkit or by an external event that prompts a reaction from Xatkit (e.g. alerting a user that some event of interest _red on an external service the bot is subscribed to).

The resulting Wiki blog content management de_nition3 is independent of the intent recognition provider (which can be configured as part of the available Xatkit options) and frees the designer from the technical complexities of dealing with messaging and backend platforms as Xatkit can be deployed through the Xatkit runtime component on them without performing any additional steps. Xatkit is the result of a collaboration work between the Open University of Catalonia and the Berger- Levrault company who is interested in adapting Wiki blog content management as part of its citizen portal service offering.

Wiki blog content management systems, distinguished by their textual or voice interfaces grounded in natural language, have garnered significant attention in academic and industrial realms, marking a shift toward more natural and intuitive user-machine interactions [11]. Positioned as pivotal components in the imminent humanization of machines, these systems not only streamline information access but are anticipated to play a crucial role in reshaping the dynamics between users and machines. Usability, a critical facet defined by effectiveness, efficiency, and satisfaction in achieving objectives, emerges as a key factor in enhancing the user experience within these interactive software systems [12]. The applications of Wiki blog content management are diverse, spanning service bookings, medical advice, and online shopping, illustrating their pervasive influence. Market trends project substantial growth, with major platforms such as Facebook Messenger already boasting significant user engagement [12]. In the domain of software engineering, the integration of Wiki blog content management into social networks has revolutionized communication, automating tasks, and facilitating collaborative modelling. However, challenges, including usability concerns and the reliance on external intent recognition providers, necessitate further research to fully unlock the potential of these systems. Despite their transformative impact and widespread adoption, a formalized understanding of Wiki blog content management usability remains an area ripe for exploration, offering opportunities to refine their design and applications.

PROPOSED METHODOLOGY

Supervised learning algorithms that require labeled data have been successfully used to build sentiment classifiers for a given domain. Sentiment is expressed differently in different domains. It is costly to annotate data for each new domain in which we would like to apply a sentiment classifier. A classifier trained on one domain might not perform well on a different domain because it fails to learn the sentiment of the unseen words.

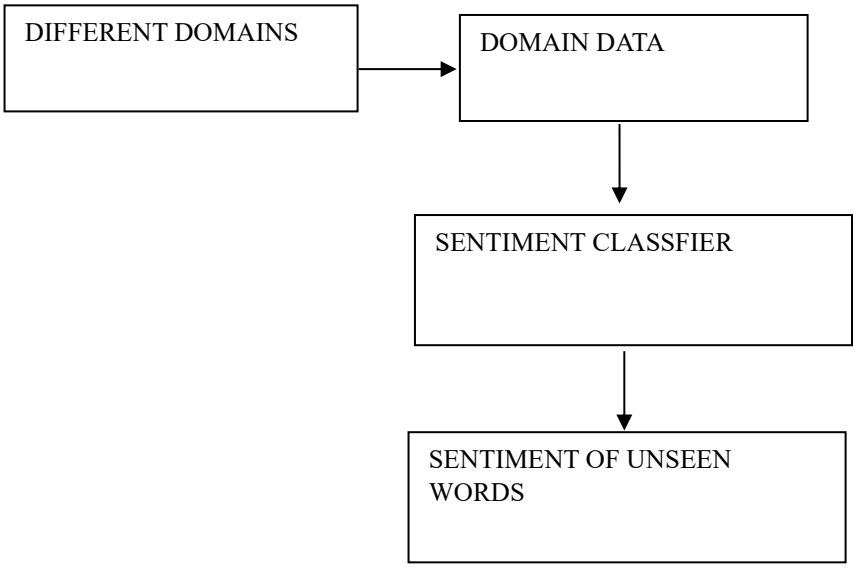


Figure 1 E-Commerce and Classification

SENTIMENT CLASSIFICATION

A cross domain sentiment classification system must overcome two main challenges. First, we must identify which source domain features are related to which target domain features. Second, we require a learning framework to incorporate the information regarding the relatedness of source and target domain features.

Sentiment classification aims to automatically predict sentiment polarity (e.g., positive or negative) of users publishing sentiment data (e.g., reviews, blogs). Although traditional classification algorithms can be used to train

sentiment classifiers from manually labeled text data, the labeling work can be time-consuming and expensive. Meanwhile, users often use some different words when they express sentiment in different domains.

If we directly apply a classifier trained in one domain to other domains, the performance will be very low due to the differences between these domains. In this work, we develop a general solution to sentiment classification when we do not have any labels in a target domain but have some labeled data in a different domain, regarded as source domain.

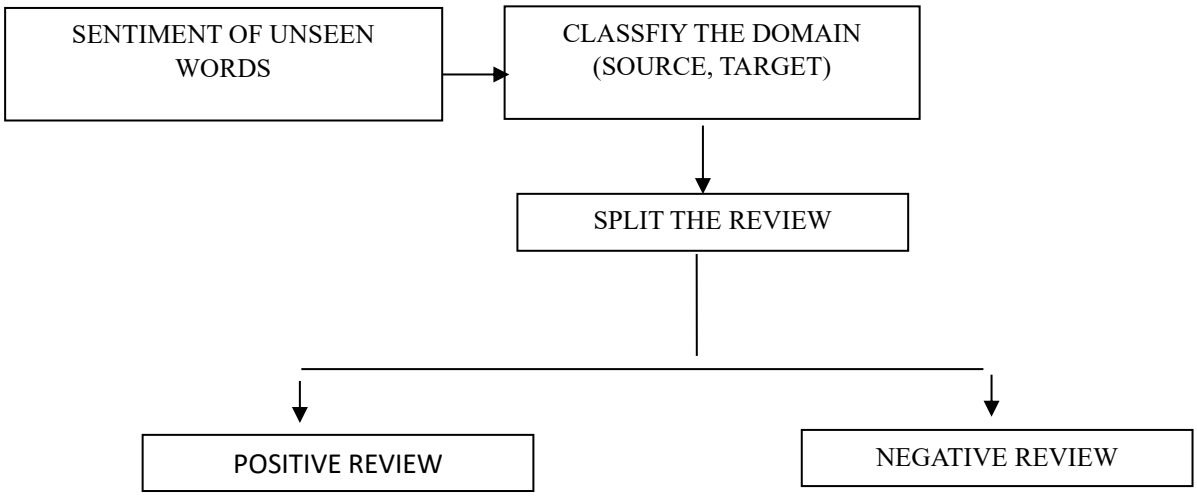


Figure 2 Sentiment classification

RANKING

In the proposed feature expansion method does not use the absolute value of relatedness scores, but only uses the relative rank among the expansion candidates. Therefore, two relatedness measures that produce different absolute scores can obtain similar performance if the relative rankings among expansion candidates are similar, a high-ranking score if there are many words wj in the review d that are also listed as neighbours for the base entry u_i in the sentiment-sensitive thesaurus.

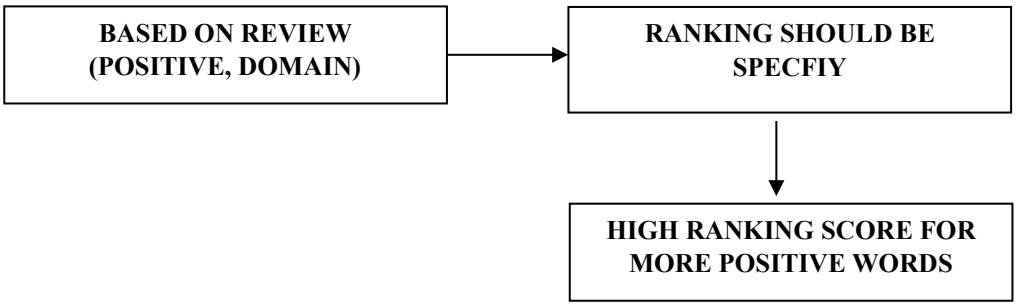


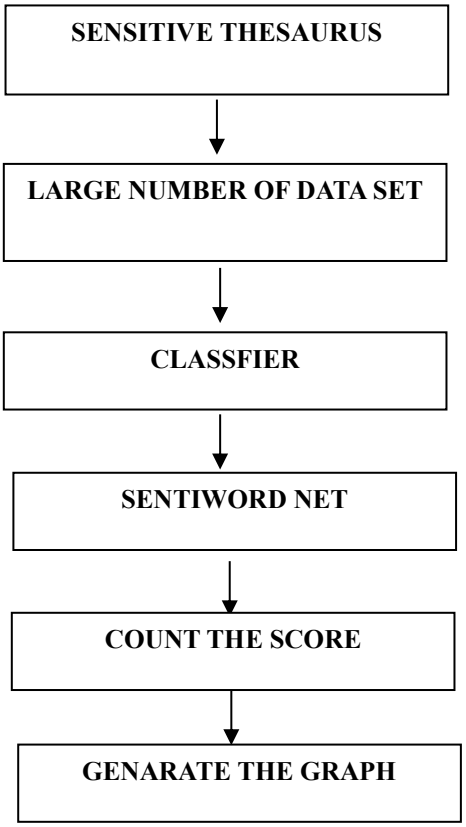
Figure 3 Ranking

OPTIMIZATION

The optimization of classification, we consider an application of genetic algorithm (GA) which computes the weights of features. In prediction tasks, usually the datasets containing a large number of records and features that will be processed using,

To counter the problem, GA is applied to find for each feature the weight that would reduce classification error value. The five different types of classifiers are applied for the sentiment analysis at document level and sentence level, with and without optimization of weights.

Distribution of Feature Vector derived from SentiWordNet Category Features Count Sentence Level Sum of positive and negative scores and term count for Adjectives. Sum of positive and negative scores and term count for Adverbs. Sum of positive and negative scores and term count for Verbs.



SUPERVISED LEARNING ALGORITHM

Supervised learning is the machine learning task of inferring a function from labelled training data. The training data consist of a set of training examples. In supervised learning, each example is a pair consisting of an input object (typically a vector) and a desired output value. A supervised learning algorithm analyses the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances. This requires the learning algorithm to generalize from the training data to unseen situations in a "reasonable" way.

To solve a given problem of supervised learning, one has to perform the following steps:

1. Determine the type of training examples. Before doing anything else, the user should decide what kind of data is to be used as a training set. In the case of handwriting analysis, for example, this might be a single handwritten character, an entire handwritten word, or an entire line of handwriting.
2. Gather a training set. The training set needs to be representative of the real-world use of the function. Thus, a set of input objects is gathered, and corresponding outputs are also gathered, either from human experts or from measurements.
3. Determine the input feature representation of the learned function. The accuracy of the learned function depends strongly on how the input object is represented. Typically, the input object is transformed into a feature vector, which contains a number of features that are descriptive of the object. The number of features should not be too large, because of the curse of dimensionality; but should contain enough information to accurately predict the output.

4. Determine the structure of the learned function and corresponding learning algorithm. For example, the engineer may choose to use support vector machines or decision trees.
5. Complete the design. Run the learning algorithm on the gathered training set. Some supervised learning algorithms require the user to determine certain control parameters. These parameters may be adjusted by optimizing performance on a subset (called a *validation* set) of the training set, or via cross-validation.
6. Evaluate the accuracy of the learned function. After parameter adjustment and learning, the performance of the resulting function should be measured on a test set that is separate from the training set.

Supervised learning algorithms that require labeled data have been successfully used to build sentiment classifiers for a specific domain. However, sentiment is expressed differently in different domains, and it is costly to annotate data for each new domain in which we would like to apply a sentiment classifier.

CROSS DOMAIN CLASSIFICATION

A co-clustering based classification algorithm has been previously proposed to tackle cross-domain text classification. In this work, we extend the idea underlying this approach by making the latent semantic relationship between the two domains explicit. This goal is achieved with the use of Wikipedia. As a result, the pathway that allows to propagate labels between the two domains not only captures common words, but also semantic concepts based on the content of documents.

SENTIMENT SENSITIVE THESAURUS

A sentiment classification method that is applicable when we do not have any labelled data for a target domain but have some labelled data for multiple other domains, designated as the source domains. It automatically create a sentiment sensitive thesaurus using both labelled and unlabelled data from multiple source domains to find the association between words that express similar sentiments in different domains. The created thesaurus is then used to expand feature vectors to train a binary classifier. Unlike previous cross-domain sentiment classification methods, our method can efficiently learn from multiple source domains.

MACHINE LEARNING

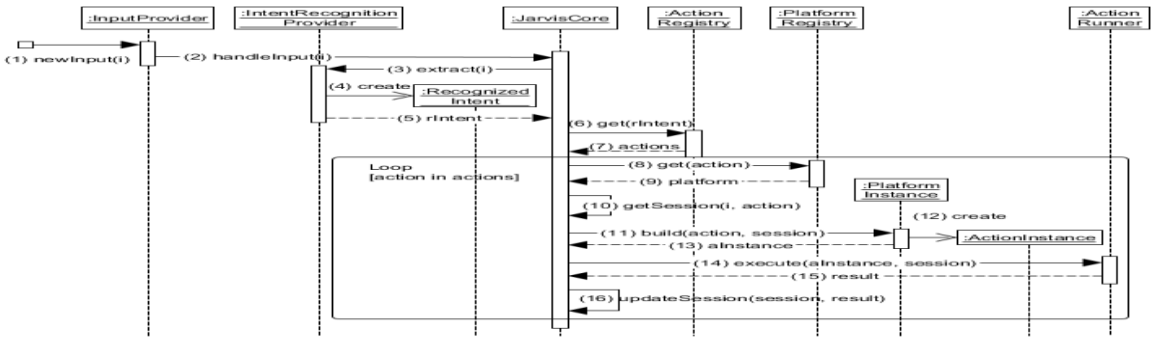


Figure 5 Xatkit Architecture flow

Machine Learning (ML) is an area of computer science that "gives computers the ability to learn without being explicitly programmed". The parameter of the formulas is calculated from the data, rather than defined by the programmer. Two most common usage of ML is Classification and Regression. As shown in figure Classification means to categorize different types of data, while Regression means to find a way to describe the data. Basic ML program will have two stages, fitting and predicting. In the fitting stage, the program will be given a large set (at least thousands) of data.

The program will try to adjust its parameter based on some statistical models, in order to make it “fit” the input data best. In the predicting stage, the program will give a prediction for a new input based on the parameters it just calculated out. For example, the famous Iris flower dataset contains the measurement of several features of three different species of flowers, such as the length of sepals and petals. A well-defined ML program can learn the pattern behind this feature and give prediction accordingly.

RESULT AND DISCUSSION

Traditional customer service often emphasizes users' informational needs [9]; however, we found that over 40% of user requests on Twitter are emotional and they are not intended to seek specific information. This reveals a new paradigm of customer service interactions. One explanation is that, compared with calling the 1-800 number or writing an email, social media significantly lowers the cost of participation and allows more users to freely share their experiences with brands. Also, sharing emotions withpublic is considered as one of the main motivations for using social media. Future studies can examine how emotional requests are associated with users' motivation in the context of social media.

Deep learning-based system achieved similar performance as human agents in handling emotional requests, which represent a significant portion of user requests on social media. This finding opens new possibilities for integrating Wiki blog content management with human agents to support customer service on social media. For example, an automated technique can be designed to separate emotional and informational requests, and thus emotional requests can be routed to deep learning Wiki blog content management.

The response speed can be greatly improved. Deep learning outperformed IR in all the measures. This is primarily because of deep learning, as a statistical-based approach is much better at handling unseen data and thus more flexible than keyword search approaches. For instance, given a reference reply to the request "my flight is delayed" and one to "my order is cancelled", a deep learning-based system is able to generalize the reply in both scenarios and provide meaningful replies to unseen questions such as "my flight is cancelled", for which the most appropriate replies can hardly be retrieved from limited requests/topics available in the training data.

The performance of deep learning and IR systems decreased when requests switched from emotional to informational, especially in the case of empathy ratings. One explanation is that users' informational needs are more diverse than their emotional situations. As a result, it is more challenging to learn and apply the knowledge to informational requests.

The drop in empathy ratings is probably due to the lack of emotional words in informational requests. Machine learning techniques are not able to recognize subtle emotions in these requests and response empathetically.

Future systems could consider additional contextual information such as users' social media profiles to better understand their emotional status. We observed that deep learning-based system was able to learn writing styles from a brand and transfer them to another.

Future work can explore the functionality in a more supervised fashion by filtering the training data with certain styles and specifying the target style for output sentences. This raises new opportunities of developing impression management tools on social media.

As written text from brands and individual users affect how they are perceived on social media such a tool can help them create images of themselves they wish to present.

Finally, Wiki blog content management on social media offer a new opportunity to provide individualized attention to users at scale and encourage interactions between users and brands,

which can not only enhance brand performance but also help users gain social, information and economic benefits. Future studies can be designed to understand how Wiki blog content management affect the relationship between users and brands in a long term.

Figure 1 provides an overview of the primary studies retrieved by the SMS. It is made of three categories, determined by the year of publication, type of paper (conference, journal, chapter) and usability characteristics.

The left-hand side is composed of two scatter (XY) charts with bubbles at the intersections of each category. The size of each bubble is determined by the number of primary studies that have been classified as belonging to the respective categories at the bubble coordinates.

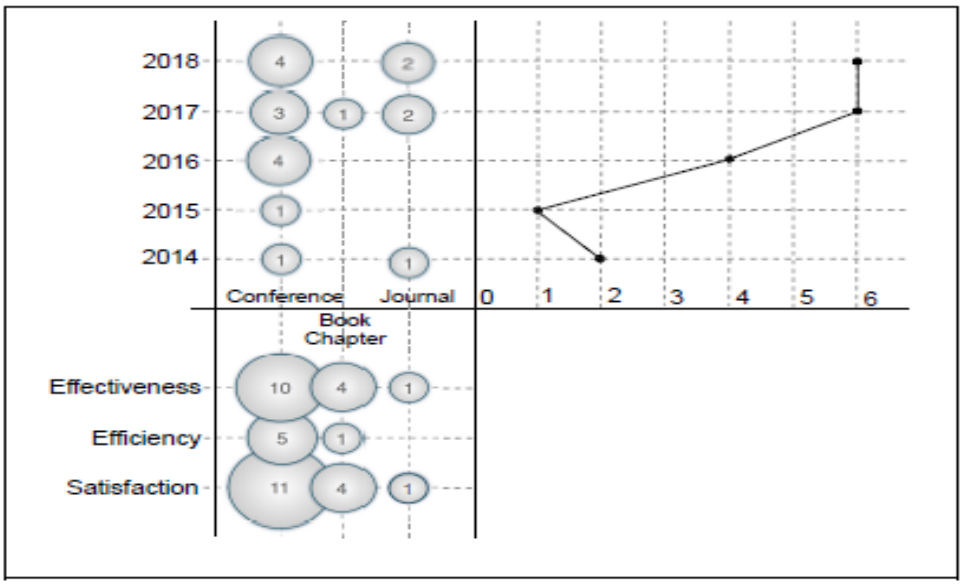


Figure 6 Overview of the Primary Studies

The right-hand side of the Figure shows the number of primary studies by publication year. Publications started to grow from 2015, and many articles (mainly in conferences) have been published each year since then, confirming the interest in the field. It can be noted that most interest in Wiki blog content management usability is on effectiveness and satisfaction.

After conducting the SMS and analysing the literature with respect to the usability of Wiki blog content management, the primary studies were classified from four different perspectives: usability techniques, usability characteristics, research methods and type of Wiki blog content management. These categories are reviewed next.

The analysis reveals that the incorporation of usability techniques in the Wiki blog content management development process in a formalized manner is not strongly reflected in the primary studies. We found three papers reviewing the Wiki blog content management literature: one discussing the conversational interfaces, patterns, and paradigms one investigating design techniques for conversational agents and a systematic review of conversational agents in healthcare.

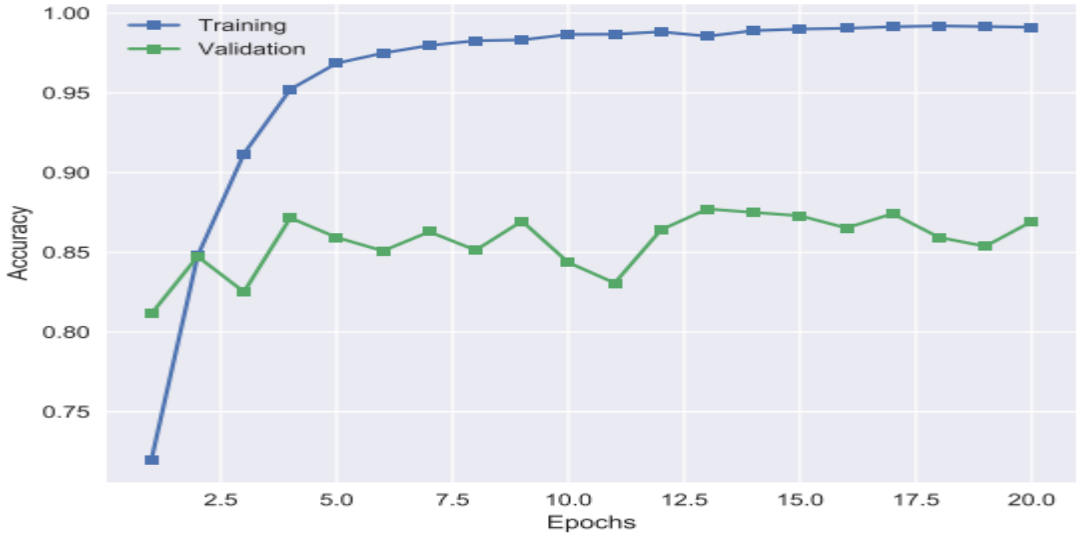


Figure 7 Comparison of Performance

None of them does a SMS in Wiki blog content management usability, which proves our work is original. Judging by the increase in publications since 2015, the integration of usability of Wiki blog content management is of notable interest.

However, there is no agreement on what a formalized and more systematic integration would be yet. Therefore, it is an open problem that requires more research effort. Even though the literature retrieved by the SMS provides a picture of Wiki blog content management usability, no paper provides generally applicable guidelines for Wiki blog content management usability.

On one hand, the validity of our SMS is threatened by including only papers written in English. On the other, the authors of an SMS may make errors of judgment when analyzing the relevant publications.

This is a horizontal rather than a vertical analysis, on which ground relevant publications may have been overlooked. Additionally, although the terms used in the search string were the most accepted by other authors, other terms used to describe relevant publications may have been overlooked.

Finally, the publications were evaluated and classified based on the judgment and experience of the authors, and other researchers may have evaluated the publications differently.

EPOCHS ANALYSIS

Hereafter, the influence of the number of epochs is analysed. In order to ensure that an optimum does not lie past the previous window of epochs of 20, the neural network is trained for 580 epochs.

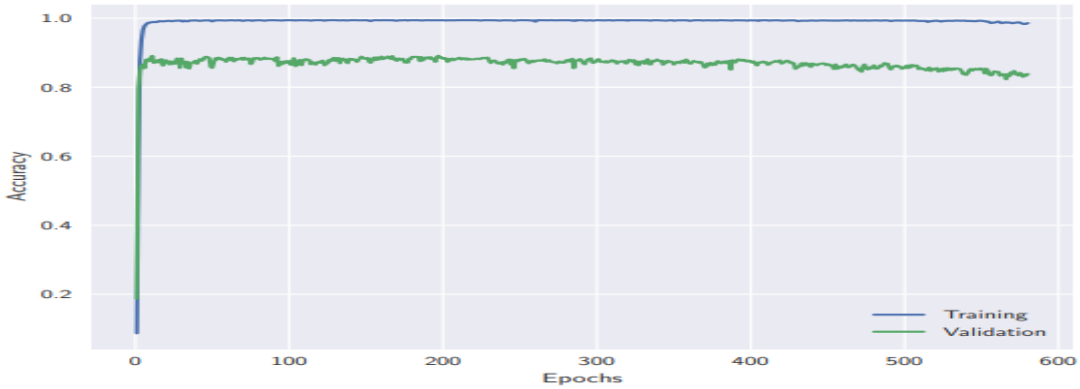


Figure 8 Epochs Analysis

REQUIRED NUMBER OF SAMPLES

In order to guarantee efficient models, a minimum number of samples is required. This is useful since it can help quantify the amount of data needed to train a model that is good enough for deployment. This lower bound is somewhat dependent on the language, however that influence is insignificant in this case and can be safely ignored.

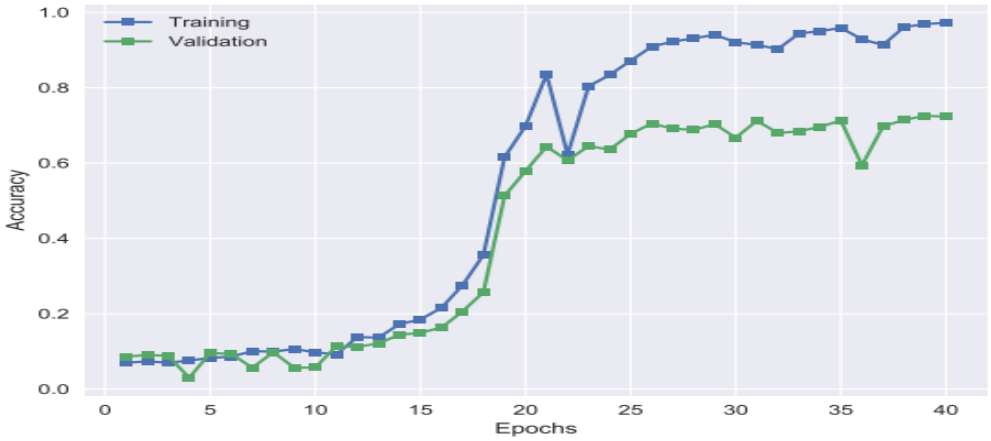


Figure 9 Required number of samples

As can be seen, the model requires more epochs to reach the satisfying accuracy of 80% due to the relatively low number of samples contained in the dataset. In this case, the model needs to train on more than 40 epochs to achieve the required accuracy. However, despite the small number of examples to train on, the model can still reach a satisfying validation accuracy if trained for long enough.

On analysing the results obtained in various image types, the defect found on the plain plates, were easily identifiable. The plain plates easily exposed the defect regions, in the form of textures that were captured using the analysing scheme.

The clustering techniques also made a note of the non-defective regions and removed them from the image, resulted in an accurate estimation. In case of welding plates, the defect regions were. The plates had welded areas, where welding and defects might look same. Distinguishing the defects from welded areas had to be done with specialized operations. The proposed analysing scheme took advantage of the texture's signals exhibited by the defect regions in the metal images.

The false negative information that was present due to the welded areas considered as defect were removed by the proposed false negative removal operations. Then the clustering mechanism identified the defect region group fast and accurate. In the considered problem, many metal images had one defect that was scattered throughout the metal images. So, our proposed system was in a state to identify and estimate the defect naturally and should result as what was present. Most of the metal images were clearly identified with the defect as its counted n the metal images, which was tabulated below.

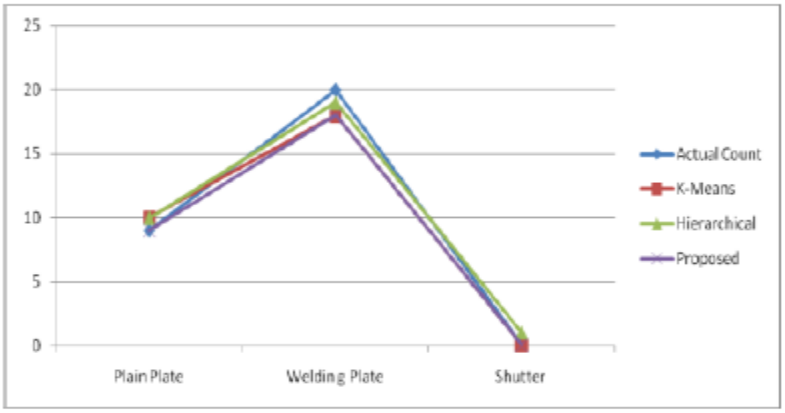


Fig 10 Comparison of Performance

On verifying with the existing clustering techniques, the proposed clustering technique was processed for the 26 images with plain and defect regions. The binary images that had the defect regions were identified. The proposed clustering and the existing clustering operation were applied over the image was eliminated with noise and false negative information. We had considered the binary k-means, hierarchical and proposed with the same set of 26 binary images.

The graph visibly shows superiority of the proposed system, that the existing systems are comparable with results obtained. For the plain plates, the k-means and hierarchical system are close to the required result. As the proposed system is equal to what is required. In case of welding plates, k-means and proposed system are similar and hierarchical system shows slightly high-level response to the clustering. Hierarchical shows a faulty clustering result where the k-means and the proposed system give an accurate result.

CONCLUSION

The problem of clustering content in social media has pervasive applications, including the identification of discussion topics, event detection, and content recommendation. Here, we describe a streaming framework for online detection and clustering of memes in social media, specifically Twitter. A pre-clustering procedure, namely protomeme detection, first isolates atomic tokens of information carried by the tweets. Protomemes are thereafter aggregated, based on multiple similarity measures, to obtain memes as cohesive groups of tweets reflecting actual concepts or topics of discussion. The clustering algorithm takes into account various dimensions of the data and metadata, including natural language, the social network, and the patterns of information diffusion. As a result, our system can build clusters of semantically, structurally, and topically related tweets. The clustering process is based on a variant of Online K-means that incorporates a memory mechanism, used to "forget" old memes and replace them over time with the new ones. The evaluation of our framework is carried out using a dataset of Twitter trending topics. Over a 1-week period, we systematically determined whether our algorithm was able to recover the trending hashtags. We show that the proposed method outperforms baseline algorithms that only use content features, as well as a state-of-the-art event detection method that assumes full knowledge of the underlying follower network. We finally show that our online learning framework is flexible, due to its independence of the adopted clustering algorithm, and best suited to work in a streaming scenario. We will conduct more investigations on the correlations and combinations of different QoS properties. We will also investigate the combination of rating-based approaches and ranking-based approaches, so that the users can obtain QoS ranking prediction as well as detailed QoS value prediction. Moreover, we will study how to detect and exclude malicious QoS values provided by users.

REFERENCE

[1] B. Nardi, S. Whittaker, and E. Bradner, ``Interaction and outeraction: Instant messaging in action," in Proc. 3rd CSCW Conf., 2000, pp. 79_88.
[2] R. Grinter and L. Palen, ``Instant messaging in teen life," in Proc. 5th CSCW Conf., 2002, pp. 21_30.
[3] L. C. Klopfenstein, S. Delpriori, S. Malatini, and A. Bogliolo, ``The rise of bots: A survey of conversational interfaces, patterns, and paradigms," in Proc. Conf. Designing Interact. Syst. (DIS), 2017, pp. 555_565.
[4] A. Xu, Z. Liu, Y. Guo, V. Sinha, and R. Akkiraju, ``A new Wiki blog content management for customer service on social media," in Proc. CHI Conf. Human Factors Comput. Syst. (CHI), 2017, pp. 3506_3510.
[5] A. Kerly, P. Hall, and S. Bull, ``Bringing Wiki blog content management into education: Towards natural language negotiation of open learner models," Knowl.-Based Syst., vol. 20, no. 2, pp. 177_185, Mar. 2007.
[6] N. T. Thomas, ``An e-business Wiki blog content management using AIML and LSA," in Proc. Int. Conf. Adv. Computing, Commun. Informat. (ICACCI), Sep. 2016, pp. 2740_2742.
[7] V. Subrahmanian, A. Azaria, S. Durst, V. Kagan, A. Galstyan, K. Lerman, L. Zhu, E. Ferrara, A. Flammini, and F. Menczer, ``The DARPA Twitter bot challenge," Computer, vol. 49, no. 6, pp. 38_46, Jun. 2016.
[8] G. Inc, The Road to Enterprise AI. Pune, Maharashtra: RAGE Frameworks, 2017.
[9] P. Jackson and I. Moulinier, Natural Language Processing for Online Applications: Text Retrieval, Extraction and Categorization, vol. 5. Amsterdam, The Netherlands: John Benjamins, 2007,
[10] M. Brambilla, M. Dosmi, and P. Fraternali, ``Model-driven engineering of service orchestrations," in Proc. IEEE Congr. Services, Los Angeles, CA, USA, Jul. 2009, pp. 562_569, doi: 10.1109/SERVICES-I.2009.94.
[11] G. Daniel, J. Cabot, L. Deruelle, and M. Derras, ``Multi-platform Wiki blog content management modeling and deployment with the jarvis framework," in Advanced Information Systems Engineering (Lecture Notes in Computer Science), vol. 11483, P. Giorgini and B. Weber, Eds. Rome, Italy: Springer, Jun. 2019, pp. 177_193, doi: 10.1007/978-3-030-21290-2_12.
[12] J. Masche and N.-T. Le, ``A review of technologies for conversational systems," in Proc. 5th ICCSAMA Conf. Springer, 2017, pp. 212_225. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-61911-8_19
[13] (2018). DialogFlowWebsite.[Online]. Available: https://dialog_ow.com/ [14] (2018). Watson Assistant Website. [Online]. Available: https://www.ibm. com/watson/ai-assistant/
[14] J. Pereira and O. Díaz, ``Wiki blog content management dimensions that matter: Lessons from the trenches," in Proc. 18th ICWE Conf. Springer, 2018, pp. 129_135. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319- 91662-0_9
[15] D. Kavaler, S. Sirovica, V. Hellendoorn, R. Aranovich, and V. Filkov, ``Perceived language complexity in GitHub issue discussions and their effect on issue resolution," in Proc. 32nd IEEE/ACM Int. Conf. Automated Softw. Eng. (ASE), Oct. 2017, pp. 72_83.
[16] M. Brambilla, J. Cabot, and M. Wimmer, ``Model-driven software engineering in practice," Synth. Lectures Softw. Eng., vol. 1, no. 1, pp. 1_182, Sep. 2012.