International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No.(02), March 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

DETECTING MALWARES AND SEARCH RANK FRAUD IN GOOGLE SEARCH USING RABIN KARP ALGORITHM

Keerthana.B[1], Sivashankari.K[2], Shaistha Tabasum.S[3]

[1,3]Department of computer science and engineering

[2]Assistant Professor

Department of Computer Science and Engineering

Dhaanish Ahmed College of Engineering

ABSTRACT

To develop a java application to search web and capture the result page for study. The organic SEO websites to be examined to find out any Fraudulent or immoral ways are whether used by the SEO specialists to promote the website and to suggest the name of such websites for removal, as we cannot remove those websites but we can suggest in an acceptable way with proven research of those websites. The landing page of all websites, the top 10 first given word are to be stored in a file using java application and such websites must be examined with on page SEO techniques used. The algorithm used is ***RABIN KARP ALGORITHM****. Java provides most powerful API's like IO a net to do coding related to internet and IO activities like reading, writing and searching the file, counting the keywords, matching.*

Keywords— Data Mining, Rabin Karp Algorithm, SQL database, JAVA.

I. INTRODUCTION

We seek to identify both malware and search rank fraud subjects in Google Play. This combination is not arbitrary: we posit that malicious developers resort to search rank fraud to boost the impact of their malware.

Fair Play, a novel system that discovers and leverages traces left behind by fraudsters, to detect both malware and apps subjected to search rank fraud. Fair Play correlates review activities and uniquely combines detected review relations with linguistic and behavioral signals gleaned from Google Play app data (87 K apps, 2.9 M reviews, and 2.4M reviewers, collected over half a year), in order to identify suspicious apps. Fair Play achieves over 95 percent accuracy in classifying gold standard datasets of malware, fraudulent and legitimate apps.

Prime objective of the current project is to study the web page source Code and to analyze the SEO Techniques used in that landing page to get importance in Google Search. We read the source and store it data base for future reference and study as Google search results may change time to time.

Search engine optimization (SEO) is the process of improving the volume and quality of traffic to a web site from search engines via "natural" ("organic" or "algorithmic") search results."The benefits generated by a well planned SEO campaign are numerous.

It gives your website exposure and visibility by improving its rankings organically without having to pay the Search Engines, SEO ensures your site is easily navigated and ensures the website is geared towards your visitors to turn them into customers, and there is no point in getting your customers to the site if they don't want to stay.

We try to develop an application using the Java Technology to read the search results of Google for a given keyword say 'Top 10 private engineering colleges in Chennai'.

To read the search results. Ignore the websites promoted through Ads and considering the top 5 organic listing. Visit the landing page of the top we page and read the HTML source code Count the key words And take a decision based on number of times the subject key word was presented. To get top rank, the ways the tag are used We seek to identify both malware and search rank fraud subjects in Google Search

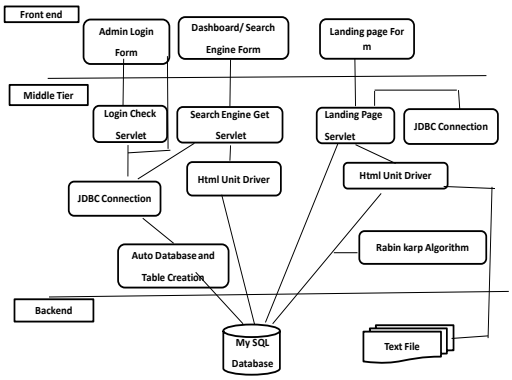
.This combination is not arbitrary: we posit that malicious developers resort to search rank fraud to boost the impact of their malware..

1.1 Contributions We propose , a system that leverages the above observations to efficiently detect Google Play fraud and malware our major contributions are: A Fraud and Malware Detection Approach. To detect fraud and malware, we propose behavioral and linguistic features, that we use Rabin Karp algorithms

1.2 Results

By using this algorithm it first Study web page source Code and then analyze SEO Techniques. SEO is the Google's Algorithm. It helps us to understand the code. After this process it goes to the landing page in that place we have lot of links are present and the links are stored in data base for future reference. The Google search results which change time to time and count the keywords. Tags used by SEO are <title>, <meta>, <Header> etc. Then we can give suggestion to Google to remove the Fake activities.

I1. PROCESS DIAGRAM



III.WHAT RABIN KARP ALGORITHM STATES

The Rabin-Karp algorithm is a string searching algorithm that uses hashing to find patterns in strings. A string is an abstract data type that consists of a sequence of characters. Letters, words, sentences, and more can be represented as strings.

String matching is a very important application of computer science. If you've ever searched through a document for a

In this module a Java program is written to count the key word in the source code of the Landing page. All the five links will show 0 before clicking the count button And after clicking the count button the field is updated With number of times the key word is found. This is very important to analyze the SEO Technics…

IV. HOW ITS WORKS

Being a web project that too being an online, we need authentication to get into landing page or Dash Board. The user id and password are hard coded, that is written inside code itself. No database is used as we have only one user.

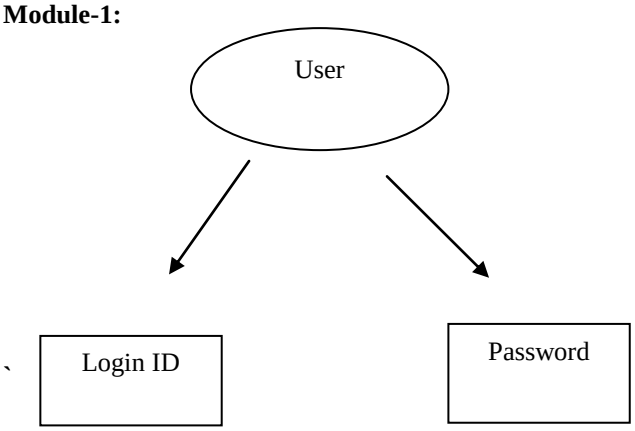
This is the prime module developed using java servlet to search the web, a Google search simulation program. The result page of the Google is saved in your folder and displayed as a result page for the user. The top five links are stored in mysql database for further research they involve JDBC connectivity .

This is the key module developed using Java Servlet to Visit the landing page or Dash Board and read the HTML Source code. The source codes are saved in a file as well in MySql Database for further reference. JDBC code is required and Selinium web driver is also used in Eclipse IDE that plays a vital role in 2nd and this module

In this module a Java program is written to count the key word in the source code of the Landing page. All the five links will show 0 before clicking the count button And after clicking the count button the field is updated With number of times the key word is found. This is very important to analyze the SEO Technics…

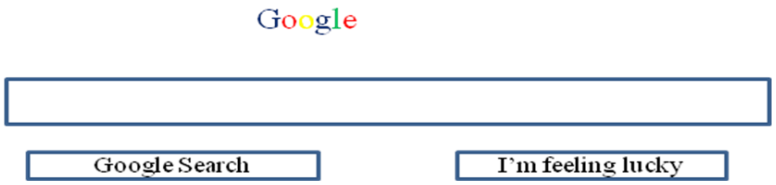
Here we write a Java Program to search the website to find out the string Like 'on the basis of ' key word. If no such key word found in that page then the website will be treated as biased or less helpful. Suggestions to Google are planned as Google always welcome suggestions from anyone.

Module-1:

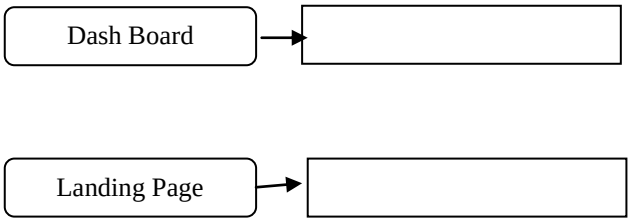


International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No.(02), March 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

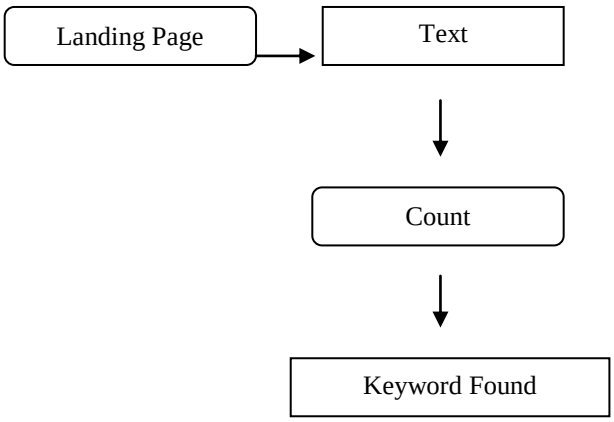
Module-2



Module-3



Module=-4



V. CONSTRUCTION

Initially we create a login form which consists of user name and password. Once you have logged into the form then you are able to enter into the webpage. In webpage there are two default options like landing page and dashboard options which is used to see the search results. The landing page consists of list of links which are searched earlier. These links are used to see who are all involved in immoral acitivities.these links are stored in sql database using landing page servlet and jdbc connection .

The dashboard which displays the keywords that are used to highlight that link in top. we use rabin karp algorithm to identify the duplication of link or fake links. In rabin karp algorithm we use string function is used to identify fake links. The Rabin-Karp algorithm is a string searching algorithm.. It uses hashing to find patterns in strings. It can also be used to detect plagiarism by comparing strings in document with strings in document .A practical application of the algorithm is detecting plagiarism.

After detecting the plagiarism we suggest the Google to take our suggestion about that link. Since the link contains some malware activities which could be ignored by the Google to have a better search links for the Google users.

VI. APPLIED ALGORITHM

e h_p(for pattern p)

Compute h_t(for the first substring of t with m length)

For i= 1 to n − m

If $h_p = h_t$

Match t[i....i+ m] with p, if matched return 1 Else

$$h_t = (d\left(h_t - t\left[i+1\right].d^{m-1}\right) + \ t\left[m+i+1\right]) \bmod q$$

End

Suppose, t= 2359023141526739921 and p = 31415,

Now, h_p= 7 (31415 = 7 (mod 13))

Substring beginning at position 7 = valid match

VII. STRING MATCHING

Rabin-Karp string searching algorithm calculates a numerical (hash) value for the pattern p,and for each m-character substring of text t. Then it compares the numerical values instead of comparing the actual symbols. If any match is found, it compares the pattern with the sub stringby naive approach. Otherwise it shifts to next substring of t to compare with p.We can compute the numerical (hash) values using Horner's rule.

Lets assume, h_0 = k

h_1 = d

□

k □ p [1] :$d_{m□1}$

–

+ p [m + 1]

Suppose, we have given a text t = [3, 1, 4, 1, 5, 2] and m = 5, q = 13;

t_0 = 31415

So t_1 = 10(31415 - $10_{5□1}$.t[1]) + t[5+1]

= 10(31415 □ 10_4:3) + 2

= 10(1415) + 2 = 14152

Here p and substring t_i may be too large to work with conveniently. The simple solution is,

we can compute p and the t_i modulo a suitable modulus q.

So for each i,

h_{i+1} = (d

□

h_i □ t [i + 1] :$d_{m□1}$

–

+ t [m + i + 1]) mod q

The modulus q is typically chosen as a prime such that d.q _ts within one computer word.

Algorithm

Compute h_p (for pattern p)

Compute h_t (for the _rst substring of t with m length)

For i = 1 to n □ m

If $h_p = h_t$

Match t[i : : : :i + m] with p, if matched return 1

Else

h_t = (d

□

h_t □ t [i + 1] :$d_{m□1}$

–

+ t [m + i + 1]) mod q

End

Suppose, t= 2359023141526739921 and p = 31415,

Now, h_p = 7 (31415 = 7 (mod 13))

substring beginning at position 7 = valid match

This algorithm has a signi_cant improvement in average-case running time over naive ap-proach.

VIII. ADVANTAGES

1. To find out the content quality in Google.
2. To help in finding relevancy of landing page.
3. To help Google's search algorithm for better accuracy

IX. CONCLUSION & FUTURE ENHANCEMENT

Internet and websites now plays predominant role in Business and all walks of life. Google search becomes a popular way of searching the we Hence every one having website would like to see his website on the top Ranks This leads to chances of

growing SEO techniques.

When people search for your products and services, you obviously want to appear as high in the search engine rankings as possible.

So we try to analyze the source code of that web page to find out the seo techniques used and try to suggest fraudulent ones. We can't change the results but can suggest such websites. .

REFERENCES

[1] Google Play. [Online]. Available: https://play.google.com/

[2] E. Siegel, "Fake reviews in Google Play and Apple App Store," Appentive, Seattle, WA, USA, 2014.

[3] Z. Miners. (2014, Feb. 19). "Report: Malware-infected Android apps spike in the Google Play store," PC World. Available: http:// www.pcworld.com/article/2099421/report-malwareinfectedandroid-apps-spike-in-the-google-play-store.html

[4] S. Mlot. (2014, Apr. 8). "Top Android App a Scam, Pulled From Google Play," PCMag. Available: http://www.pcmag.com/ article2/0,2817,2456165,00.asp

[5] D. Roberts. (2015, Jul. 8). "How to spot fake apps on the Google Play store," Fortune. Available: http://fortune.com/2015/07/08/ google-play-fake-app/

[6] Freelancer. [Online]. Available: http://www.freelancer.com

[7] Fiverr. [Online]. Available: https://www.fiverr.com/

[8] BestAppPromotion. [Online]. Available: www.bestreviewapp. com/

[9] G. Wang, et al., "Serf and turf: Crowdturfing for fun and profit," in Proc. ACM WWW, 2012. [Online]. Available: http://doi.acm. org/10.1145/2187836.2187928

we propose to extract domain-specific sentiment knowledge by combining limited labeled samples with massive unlabeled samples, which is not considered in previous work. The domain-specific sentiment knowledge contains rich specific sentiment expressions used in each domain and can provide important prior information for learning domain-specific sentiment classifiers. It is also used in our approach to measure the similarities between different domains. Third, a large multi-domain sentiment dataset was added to the experiments to evaluate the performance of our approach more thoroughly. In addition, more experiments were conducted. For example, we conducted experiments to explore the influence of training data size on the performance of our approach to verify whether our approach can handle the problem of scarce labeled data by training sentiment classifiers for multiple domains collaboratively. We also conducted experiments to evaluate the time complexity of the proposed parallel algorithm and compare it with the single-node version algorithm . Besides, more detailed analysis and discussions on the experimental results are presented in this paper. Thus, compared with the previous version work [20], a large amount of new content has been added to this paper.The rest of this paper is organized as follows. In Section 2, we briefly review several representative related works. In Section3, we introduce two important components in our approach , i.e., domain-specific sentiment knowledge extraction and domain similarity measure. In Section 4, we present our collaborative multi-domain sentiment classification approach as well as the optimization algorithms in detail. In Section 5, we report the experimental results on benchmark multi-domain sentiment datasets.In Section 6, we conclude this paper.

2. RELATED WORK

In this section, we briefly review several representative works on multi-domain sentiment classification and multi task learning.

2.1 Multi-Domain Sentiment Classification

Sentiment classification has been widely known as a highly

domain-dependent problem [13], [14], [15], [21], [22]. Different domains have different ways to express sentiments, and a sentiment classifier trained in one domain usually perform not very well in another domain.For example, “easy” is a positive word in Kitchen domain (e.g., “this fryer is very easy to use”). However, it is frequently used as a negative word in Movie domain (e.g., “the ending of this film is easy to guess”). Thus, the sentiment classifier trained in Movie domain cannot predict the sentiment of “easy” in Kitchen domain accurately. An intuitive method to solve this problem is training a domain specific sentiment classifier or building a domain-specific sentiment lexicon for each domain independently [10], [23]. For example, Pang etal. built sentiment classifiers for movie reviews using machine learning techniques such as SVM and Naive Bayes based on the labeled data of this domain [10]. Lu et al. proposed to construct a domain-specific sentiment lexicon by incorporating information from various sources in this domain, such as sentiment labels and linguistic heuristics [23]. However, in many domains, the labeled data is usually in limited size and insufficient to extract accurate and robust sentiment information. In addition, since thereare massive domains involved in online user genrated content, it is expensive and time-consuming to manually annotate enough samples for each domain.A popular method to reduce the effort of manual annotation is using transfer learning to adapt the sentiment classifier from a source domain with sufficient labeled data to a target domain with scarce or no labeled data [24]. Many cross domain sentiment classification methods belong to this kind [13], [14], [22], [25]. For example, Blitzer et al. proposed a sentiment domain adaption method based on structural correspondence learning (SCL) algorithm [13]. The core idea of SCL is finding correspondence among features from different domains by computing their associations with pivot features. Pan et al. proposed a spectral feature alignment (SFA) algorithm for cross domain sentiment classification to reduce the gap of sentiment expressions from different domains [22]. He etal. proposed to extract polarity-bearing topics based on a modified joint sentiment-topic (JST) model using data from both source and target domains [25]. These topics are used to augment the feature representations of texts from both domains. Then a sentiment classifier is trained on labeled data in source domain and applied to unseen data in target domain. The assumption behind these cross domain sentiment classification methods is that there is sufficient labeled data in source domain while the labeled data in target domain is scarce or non-existent [16]. The goal of these methods is to adapt the sentiment knowledge extracted from the labeled data of source domain to target domain. However, in this paper we assume that the labeled data in each domain is insufficient, and our goal is to train an accurate and robust sentiment classifier for each domain in a collaborative way by exploiting the sentiment relatedness among these domains. Another line of research in multi-domain sentiment classification is sentiment classifier combination [15], [21]. For example, Liet al. proposed to combine the classification results of sentiment classifiers trained in different domains to make final predictions[21]. These methods can be regarded as integrating the sentiment knowledge from different domains at the classification stage, while in our approach the sentiment knowledge from different domains is shared at the learning stage in order to help train sentiment classifiers for each domain more accurately when labeled data is in sufficient. The experimental results validate that our approach is more effective in exploiting the sentiment knowledge of multiple domains than these sentiment classifier combination methods.

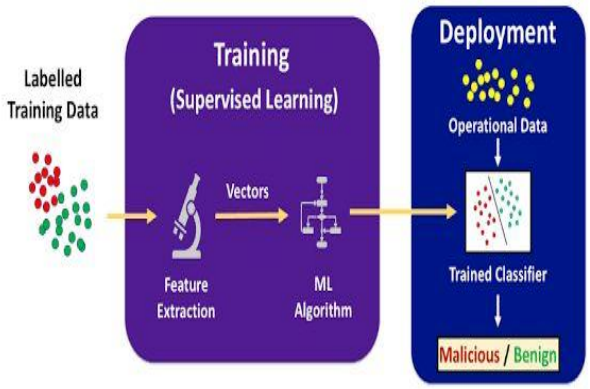
2.2 Multi-Task Learning

The approach proposed in this paper is based on multi-task

learning [26], [27]. The aim of multi-task learning is to improve the generalization ability and prediction performance by learning multiple related tasks simultaneously and leveraging the common knowledge shared by these tasks appropriately [27]. The main difference between different multi-task learning methods lies in how they model and incorporate the task relatedness [27]. For example, Evgeniou and Pontil proposed a regularized multi-task learning method [26]. In their method, the classification models of related tasks are constrained to be similar with their average model. Thus, in this method the relatedness between various tasks is introducedby the average model and the direct relations between these tasks are not taken into consideration. Liu et al. proposed a multi-task feature learning method [28]. In their method, the classification models of related tasks are assumed to share the same sparse feature space, which is selected by group Lasso [29]. However, this assumption may not hold in multi-domain sentiment classification scenario, since different features are used to express sentiments in different domains. In trace-norm regularized multi-task learning methods [30], [31], the models from multiple related tasks are assumed to share a low-dimension A popular method to reduce the effort of manual annotation is using transfer learning to adapt the sentiment classifier from a source domain with sufficient labeled data to a target domain with scarce or no labeled data [24]. Many cross domain sentiment classification methods belong to this kind [13], [14], [22], [25]. For example, Blitzer et al. proposed a sentiment domain adaption method based on structural correspondence learning (SCL) algorithm [13]. The core idea of SCL is finding correspondence among features from different domains by computing their associations with pivot features. Pan et al. proposed a spectral feature alignment (SFA) algorithm for cross domain sentiment classification to reduce the gap of sentiment expressions from different domains [22]. He etal. proposed to extract polarity-bearing topics based on a modified joint sentiment-topic (JST) model using data from both source and target domains [25]. These topics are used to augment the feature representations of texts from both domains. Then a sentiment classifier is trained on labeled data in source domain and applied to unseen data in target domain. The assumption behind these cross domain sentiment classification methods is that there is sufficient labeled data in source domain while the labeled data in target domain is scarce or non-existent [16]. The goal of these methods is to adapt the sentiment knowledge extracted from the labeled data of source domain to target domain. However, in this paper we assume that the labeled data in each domain is insufficient, and our goal is to train an accurate and robusst sentiment classifier for each domain in a collaborative way by exploiting the sentiment relatedness among these domains. Another line of research in multi-domain sentiment classification is sentiment classifier combination [15], [21]. For example, Liet al. proposed to combine the classification results of sentiment classifiers trained in different domains to make final predictions [21]. These methods can be regarded as integrating the sentiment knowledge from different domains at the classification stage, while in our approach the sentiment knowledge from different domains is shared at the learning stage in order to help train sentiment classifiers for each domain more accurately when labeled data is insufficient. The experimental results validate that our approach is more effective in exploiting the sentiment knowledge of multiple domains than these sentiment classifier combination methods.

2.2 Multi-Task Learning

The approach proposed in this paper is based on multi-task learning [26], [27]. The aim of multi-task learning is to improve the generalization ability and prediction performance by learning multiple related tasks simultaneously and leveraging the common knowledge shared by these tasks appropriately [27]. The main difference between different multi-task learning methods lies in how they model and incorporate the task relatedness [27]. For example, Evgeniou and Pontil proposed a regularized multi-task learning method [26]. In their method, the classification models of related tasks are constrained to be similar with their average model. Thus, in this method the relatedness between various tasks is introduced by the average model and the direct relations between these tasks are not taken into consideration. Liu et al. proposed a multi-task feature learning method [28]. In their mehod, the classification models of related tasks are assumed to share the same sparse feature space, which is selected by group Lasso [29]. However, this assumption may not hold in multi-domain sentiment classification scenario, since different features are used to express sentiments in different domains. In trace-norm regularized multi-task learning methods [30], [31], the models from multiple related tasks are assumed to share a low-dimensional subspace. In clustered multitask learning methods [32], [33], the group structure of models from various tasks is explored. The models of tasks from the same cluster are constrained to be more similar with each other than those from different clusters. Different from above multi-task learning methods, in our collaborative multi-domain sentiment classification approach, the task relatedness is modeled in two aspects. First, the sentiment classification models of multiple domains share the same global component. The classification model of each domain can contribute to this global component and benefit from it during the learning stage. Second, each pair of domain-specific sentiment classification models is linked via their domain similarity and learned collaboratively. Similar domains are encouraged to share more sentiment information with each other than dissimilar domains. In addition, in our approach we propose to extract prior general sentiment knowledge from general-purpose sentiment lexicons to guide the learning of the global sentiment classifier, and extract domain-specific sentiment knowledge from both labeled and unlabeled data to enhance the learning of the domain-specific sentiment classifiers. In these ways, the shared sentiment information among different domains is fully exploited and the problem of scarce labeled data is effectively alleviated. Thus, our approach is more suitable for multi-domain sentiment classification than these state-of-the-art multi-task learning methods.



3 DOMAIN-SPECIFIC SENTIMENT KNOWLEDGE AND DOMAIN SIMILARITY

In this section, we introduce two important components that will be used in our collaborative multi-domain sentiment classification approach (CMSC). The first one is the domain-specific sentiment knowledge, which is mined from massive unlabeled samples and a small number of labeled samples. It can provide prior knowledge of the sentiment expressions used in each domain. The second one is domain similarity, which measures whether two domains share similar terms and sentiment expressions.

3.1 Domain-Specific Sentiment Knowledge Extraction

Every domain has many domain-specific sentiment expressions, which are not captured by general-purpose sentiment lexicons or sentiment datasets of other domains. For example, "quick" is a positive word in Kitchen domain (e.g., "It's a quick and quiet way to clean up"). However, it is a neutral word in many sentiment lexicons, such as MPQA1 [34]. Another example is "easy", which is a positive word in Kitchen domain (e.g., "Hand washing is easy and quick") but frequently conveys negative sentiment in Movie domain (e.g., "The ending of this film is easy to guess"). Thus, we propose to extract domain-specific sentiment knowledge from the data of a specific domain. It is formulated as the sentiment expression distribution of this domain and can provide prior knowledge for learning domain-specific sentiment classifiers. Two kinds of data are combined to extract domain-specific sentiment knowledge for each domain. The first kind of data is the labeled samples, which are associated with sentiment labels and can be used to infer domain-specific sentiment expressions directly. A common observation in sentiment analysis field is that the words occur more frequently in positive samples than negative The textual content based domain similarity is motivated by the observation that although different topics and opinion targets are discussed in different domains, similar domains may share many common terms. For example, in both Smart Phone and Digital Camera domains, terms like "screen", "battery", and "image" are frequently used. In contrast, the probability of two far different
domains such as Smart Phone and Book sharing many common terms is low. Thus, we propose to measure the similarity between domains based on their textual content.

Inspired by the work in [38], here we select Jensen-Shannon divergence to measure the similarity of two domains based on their textual term distributions. Denote dm 2 RD_1 and dn 2 RD_1 as the term distribution vectors of domains m and n respectively, where D represents the dictionary size. Dmt 2 [0; 1] stands for the probability of term t occurring in domain m. Then the textual content based domain similarity between domains m and n is formulated as:

ContentSim(m; n) =1 - DJS(dm || dn)
=1 -1/ 2 (DKL(dm||d) + DKL(dn || d));

where d = 12 (dm + dn) is the average distribution, DJS(_) represents Jensen-Shannon divergence, and DKL(_) is the Kullback- Leibler divergence which is defined as:

DKL(p || q) =XDt=1p(t) log2p(t)/q(t)

Since the base of logarithm used in Eq. (5) is 2, DJS(dmjjdn) 2[0; 1]. Thus, the range of the textual content based domain similarity defined in Eq. (4) is also [0; 1].

3.2.2 Sentiment Expression Based Domain Similarity

The textual content based domain similarity introduced in previous subsection can measure whether two domains have similar word usage patterns. However, high similarity in textual content does not necessarily mean that sentiment words are used in similar ways in these domains. For example, both CPU and Battery belong to electronic hardware. In CPU domain, the word "fast" is usually positive. For instance, "Intel Core i7 is very fast." However, in Battery domain, the word "fast" is frequently used as a negative word (e.g., "This battery runs out too fast"). Thus, measuring domain similarity based on sentiment expressions may be more suitable for multi-domain sentiment classification task. Denote pm and pn as the sentiment word distributions of domains m and n respectively, which are extracted from both labeled and unlabeled samples according to previous subsection.Then the sentiment expression based domain similarity between domains m and n is defined as the cosine similarity of their sentiment word distributions:

$$\text{SentiSim}(m; n) = \frac{p^m \quad p^n}{\text{kpmk2 \_ kpnk2}}$$

Note that SentiSim(m; n) defined in Eq. (6) can be negative in theory, although the probability is very small. In this paper, we constrain that domain similarities should be non-negative. Thus, if the SentiSim score between a pair of domains is negative, then we set it to zero.

4.3.1 An Accelerated Algorithm

In this section, we introduce the FISTA based accelerated algorithm for our approach which can be conducted on a single computing node. As mentioned before, the optimization problem
in our approach is nonsmooth. Although we can use subgradient descent method to solve it, the convergence rate of subgradient method is O(1=pk) and is far from satisfactory, where k is the
number of iterations. Thus, we propose to use the accelerated
algorithm based on FISTA [18] . when f is smooth (such as squared loss and log loss).
This algorithm has the same computational complexity as gradient method and subgradient method in each iteration, and at the same time has a convergence rate of O(1=k2), much faster than that of gradient method (O(1=k)) and subgradient method (O(1=pk)).
Different from gradient method and subgradient method where current solution is computed using the last solution in each iteration, in FISTA the current solution is estimated using the last two solutions and the "momentum" between them is exploited to accelerate the optimization process [18]. In each iteration of FISTA, two kinds of points are sequentially updated. The first kind of point (denoted as search point) is a linear combination of last two solutions, which is defined as:

$v_{k+1} = w_k + a_k(w_k - w_{k-1})$;

$V_{k+1} = W_k + a_k(W_k - W_{k-1})$:

The second kind of point is the gradient update of the search point (denoted as approximate point). This point is updated using following steps. First, denote:

$$g(w;W) = \sum_{m=1}^{M}\sum_{i=1}^{N_m} f(x_i^m; y_i^m; w + W_{:m}) + \alpha_1 w^T p + \alpha_2 \sum_{m=1}^{M}(w + W_{:m})^T p_m + \sum_{m=1}^{M}\sum_{n=1}^{6=m} S_{m;n}$$
$$\|W_{:m} = W_{:n}\|_2^2 + \frac{1}{2}(k w_k^2 + k W_k^2)$$:

Then the partial derivatives of g with respect to w andWare:

$$\frac{\partial g(w,W)}{\partial w} = \sum_{m=1}^{N} \frac{\partial}{\partial w} f(x_i^m, y_i^m, w + W_{\cdot,m}) - \alpha_1 p - \alpha_2 \sum_{m=1}^{M} p_m + 2\lambda_1 w,$$

$$\frac{\partial g(w,W)}{\partial W_{\cdot,m}} = \sum_{i=1}^{N_m} \frac{\partial}{\partial W_{\cdot,m}} f(x_i^m, y_i^m, w + W_{\cdot,m}) - \alpha_2 p_m + 4\beta \sum_{n=1}^{M} S_{m,n}(W_{\cdot,m} - W_{\cdot,n}) + 2\lambda_1 W_{\cdot,m}.$$

(When f is squared loss, $\frac{@}{@w} f(x_i^m, y_i^m; w + W_{:m})$ and $\frac{@}{@W_{:m}} f(x_i^m, y_i^m; w + W_{:m})$ in Eq. (10) are both equivalent to $2((w + W_{:m})^T x_i^m - y_i^m) x_i^m$. When f is log loss,

$$\frac{@}{@w} f(x_i^m, y_i^m; w + W_{:m}) = @$$

@W_;m

f(xmi

; ym

i ;w +W_;m) =

□ym

i xmi

=

□

1 + exp(ym

i (w +W_;m)T xmi

)

–

.

Then the approximate points in the kth iteration are updated as follows:

wk+1 =S_2=Lk

–

vk+1 □

1

Lk

@g(w;W)

@w

jw=vk+1;W=Vk+1

–

;

Wk+1

_;m =S_2=Lk

–

Vk+1

_;m □

1

Lk

@g(w;W)

@W_;m

jw=vk+1;W=Vk+1

–

;

where S is the soft thresholding operator and is defined as

S_(x) = [x□_]+□[□x□_]+ [43]. 1

Lk

is the step size at the kth

iteration and its value is selected to satisfy following inequation:

$$g(w_{k+1};W_{k+1}) \_ g(v_{k+1};V_{k+1}) + \frac{L_k}{2}\|w_{k+1} \square v_{k+1}\|_2^2 + \frac{L_k}{2}\|W_{k+1} \square V_{k+1}\|_F^2 + (w_{k+1} \square v_{k+1})^T @g(w;W)/@w\,|_{w=v_{k+1};W=V_{k+1}} + \sum_{m=1}^{M}(W_{k+1}\_;m \square V_{k+1}\_;m)^T @g(w;W)/@W\_;m\,|_{w=v_{k+1};W=V_{k+1}}: \quad (11)$$

The complete accelerated algorithm for our approach (Eq. (7)) is summarized in Algorithm 1 when loss function f is smooth. Note that if the loss function f used in our approach is nonsmooth, for example, f is hinge loss, then Algorithm 1 cannot be applied to find the optimal solution. In this case, we use subgradient descent method to solve our approach.

4.3.2 A Parallel Algorithm

When the domains to be analyzed are massive, it is inefficient to train sentiment classifiers for them on a single computing node due to the limit of memory and computational ability. Motivated by [19] and [44], here we propose a parallel algorithm based on Alternating Direction Method of Multipliers (ADMM) [19] to solve our approach more efficiently.

Assume there are G parallel nodes, such as computers and CPU cores, and we partition the domains to be analyzed into G groups. The domains in the same group are processed at the same node, and different groups are processed at different nodes. Denote M_g as the set of domains in group g. We keep a copy of w in each group and denote it as v_g in group g. In addition, we also

keep a copy ofW_;m andW_;n for each pair of domains m and

Algorithm 1 The accelerated algorithm for our approach when

loss function f is smooth.

1: Input: f(xmi

; ym

i);m = 1; :::;M; i = 1; :::Nmg, p,

fpm;m = 1; :::;Mg, S, _1, _2, _, _1, _2, _ > 1, L0.

2: Output: w, W.

3: Initialize w1 = w0 = 0, W1 = W0 = 0, k = 0, L = L0.

4: while the convergence condition is not satisfied do

5: k = k + 1, ak = k

k+3 .

6: vk+1 = wk + ak(wk wk1).

7: Vk+1 =Wk + ak(Wk Wk1).

8: Compute @g(w;W)

@w jw=vk+1;W=Vk+1 (Eq. (10)).

9: wk+1 = S_2=L(vk+1 1

L

@g(w;W)

@w jw=vk+1;W=Vk+1).

10: for m = 1 to M do

11: Compute @g(w;W)

@W_;m

jw=vk+1;W=Vk+1 (Eq. (10)).

12: Wk+1

_;m =

S_2=L(Vk+1

_;m 1

L

@g(w;W)

@W_;m

jw=vk+1;W=Vk+1).

13: end for

14: while Eq. (11) doesn't hold do

15: L = _L.

16: wk+1 =

S_2=L(vk+1 1

L

@g(w;W)

@w jw=vk+1;W=Vk+1).

17: for m = 1 to M do

18: Wk+1

_;m =

S_2=L(Vk+1

_;m 1

L
@g(w;W)
@W_;m
jw=vk+1;W=Vk+1).

19: end for

20: end while

21: end while

22: w = wk+1.

23: W=Wk+1.

n, and denote them as vm;n and vn;m. Then the optimization problem in the model of our approach (Eq. (7)) is equivalent to:

min
XG
g=1
X
m2Mg
XNm
i=1
f(xmi
; ym
i ; vg +W_;m) □ _2
MX
m=1
WT
_;mpm
□ wT (_1p + _2
MX
m=1
pm) + _1(kwk22
+ kWk2
F)
+ _
MX
m=1
X
n6=m
Sm;nkvm;n □ vn;mk22
+ _2(kwk1 + kWk1;1);
s.t.: vg = w; g = 1; :::;G
vm;n =W_;m; n = 1; :::;M:

In ADMM, above optimization problem is further transformed into an augmented Lagrangian form as follows:

L(!; _; _) =
XG
g=1
X

m2Mg

X_{Nm}

i=1

f(x_{mi}

; y_m

$_i$; v_g +W_;m)

□ _2

MX

m=1

W_T

_;mp_m □ w_T (_1p + _2

MX

m=1

p_m)

+ _

MX

m=1

X

n6=m

$S_{m;n}kv_{m;n}$ □ $v_{n;m}k_{22}$

+ $_1$(kwk_{22}

+ kWk_2

F)

+ $_2$(kwk_1 + k$Wk_{1;1}$) +

_

2

X_G

g=1

(kw □ v_g + u_gk_{22}

□ ku_gk_{22}

)

+

_

2

MX

m=1

X

n6=m

(kW_;m □ $v_{m;n}$ + $u_{m;n}k_{22}$

□ k$u_{m;n}k_{22}$

);

(12)

where _ is a positive penalty coefficient, u_g 2 R_{D_1} and

$u_{m;n}$ 2 R_{D_1} are scaled dual variables. !, _, and _ are variable

IEEE Transactions on Knowledge and Data Engineering (Volume: 29, Issue: 7, July 1 2017)

8

sets, introduced to represent three kinds of variables. Among them, ! = fw;W_;m;m 2 [1;M]g, _ = fvg; vm;n; g 2 [1;G]; m; n 2 [1;M]g and _ = fug; um;n; g 2 [1;G]; m; n 2 [1;M]g. In ADMM, these three kinds of variables are updated sequentially in each iteration, which is different from traditional multiplier methods where all variables are updated simultaneously. Specifically, in the kth iteration !, _, and _ are updated as follows:

!k+1 = arg min

!

L(!; _k; _k); (13)

_k+1 = arg min

_

L(!k+1; _; _k); (14)

_k+1 = arg max

_

L(!k+1; _k+1; _): (15)

We will introduce these steps one by one in detail.

Updating !k+1. According to Eq. (12) and Eq. (13), the update processes of w and W_;m are separable. In addition, updating w can be conducted at a single node as follows:

wk+1 = arg min

w

_

2

XG

g=1

kw □ vk

g + ukg

k22

+ _1kwk22

+ _2kwk1 □ wT (_1p + _2

MX

m=1

pm):

Above optimization problem is convex but nonsmooth. Thanks to the proximal algorithm [43], we can derive an analytical solution to it as follows:

wk+1 = S _2

_G+2_1

XG

g=1

(vk
g □ ukg
) + _1p + _2
MX
m=1
pm
!
;

where S(_) is the soft thresholding operator [43].

W_;m is updated as follows:

Wk+1
_;m = arg min
W_;m
XNm
i=1
f(xmi
; ym
i ; vg +W_;m) □ _2WT
_;mpm
+
_
2
X
n6=m
kW_;m □ vk
m;n + uk
m;nk22
+ _1kW_;mk22
+ _2kW_;mk1;

(16)

where g is the group which domain m belongs to. According to Eq. (16), the updating ofW_;m;m = 1; :::;M is separable across different domains and can be computed independently. Thus we update W_;m in parallel at different nodes. However, since the optimization problem in Eq. (16) is convex but nonsmooth, and there is no analytical solution to it, it may take many iterations to find the optimal solution. Thus, we propose to solve it using FISTA [18] algorithm when the loss function f is smooth. The detailed algorithm can be derived according to the steps in previous subsection, and is omitted here due to space limit. If f is not smooth, subgradient descent method is applied to solve Eq. (16).

Updating _k+1. According to Eq. (12) and Eq. (14), the

update processes of v_g and $v_{m;n}$ are also separable and can be conducted independently. In addition, the updating of v_g; g = 1; :::;G is separable across different domain groups and can be conducted at different nodes. At the node which domain group M_g belongs to, v_g is updated as follows:

$$v^{k+1}_g = \arg\min_{v_g} \sum_{m2M_g} \sum_{i=1}^{N_m} f(x_{mi}; y^m_i; v_g + W^{k+1}_{\_;m}) + \frac{\rho}{2} \| W^{k+1} \square v_g + u^k_g \|^2_{k22} : \quad (17)$$

According to Eq. (17), updating v_g also needs to solve a convex optimization problem. Similar with updating$W_{_;m}$, we propose to use FISTA algorithm to update v_g when loss function f is smooth and apply subgradient descent method when f is nonsmooth.

From Eq. (12), the updating of $v_{m;n}$ is separable across different pairs of domains, and can be solved in a parallel way. $v^{k+1}_{m;n}$ and $v^{k+1}_{n;m}$ need to be updated jointly, and the detailed updating formulation is as follows:

$$f_{v^{k+1}_{m;n}; v^{k+1}_{n;m}} g = \arg\min_{v_{m;n}; v_{n;m}} 2\_S_{m;n} \| v_{m;n} \square v_{n;m} \|^2_{k22} + \frac{\rho}{2} \| W^{k+1}$$

_;m □ vm;n + uk
m;nk22
+
_
2
kWk+1
_;n □ vn;m + uk
n;mk22
:

It is a convex optimization problem and its analytical solution is:

vk+1
m;n = _(Wk+1
_;m + uk
m;n) + (1 □ _)(Wk+1
_;n + uk
n;m);
vk+1
n;m = (1 □ _)(Wk+1
_;m + uk
m;n) + _(Wk+1
_;n + uk
n;m);

(18)

where _ = (4_Sm;n + _)=(8_Sm;n + _).

Updating _k+1. From Eq. (12) and Eq. (15) we can see that the update processes of um and um;n are also separable. In addition, the updating of ug is separable across different domain groups and can be computed as follows:

uk+1
g = wk+1 □ vk+1
g + ukg: (19)

Besides, the updating of um;n is separable across different domain pairs and can be conducted as follows:

uk+1
m;n =Wk+1
_;m □ vk+1
m;n + uk
m;n:

CONCLUSION

This paper presents a collaborative multi-domain sentiment classification approach. Our approach can learn accurate sentiment classifiers for multiple domains simultaneously in a collaborative way and handle the problem of insufficient labeled data by exploiting the sentiment relatedness between different domains. In our approach, the sentiment classifier of each domain is decomposed into two components, a global one and a domain-specific one. The global model can capture the general sentiment knowledge shared by different domains and the domain-specific models are used to capture the specific sentiment expressions of each domain. We propose to extract domain-specific sentiment knowledge from both labeled and unlabeled samples, and use it to enhance the learning of the domain-specific sentiment classifiers. Besides, we propose to use the prior general sentiment knowledge in general-purpose sentiment lexicons to guide the learning of the global sentiment classifier. In addition, we propose to incorporate the similarities between different domains into our approach as regularization over the domain-specific sentiment classifiers to encourage the sharing of sentiment information between similar domains. A novel domain similarity measure based on sentiment word distributions is proposed. We formulate the model of our approach into a convex optimization problem. Moreover, we introduce an accelerated algorithm to solve the model of our approach efficiently, and propose a parallel algorithm to further improve its efficiency when domains to be analyzed are massive. Experimental results on benchmark datasets show that our approach can improve the performance of multi-domain sentiment classification effectively, and outperform baseline methods significantly.

ACKNOWLEDGMENTS

The authors thank the reviewers for their insightful comments and constructive suggestions on improving this work. This research is supported by the National Natural Science Foundation of China (Grant nos. U1536201, U1405254, and 61472092), the National Science and Technology Support Program of China (Grant nos. 2014BAH41B00 and 2015AA020101), and the Initiative Scientific Research Program of Tsinghua University.

REFERENCES

[1] B. Pang and L. Lee, “Opinion mining and sentiment analysis,” Foundations and trends in information retrieval, vol. 2, no. 1-2, pp. 1–135, 2008.

[2] B. Liu, “Sentiment analysis and opinion mining,” Synthesis Lectures on Human Language Technologies, vol. 5, no. 1, pp. 1–167, 2012.

[3] J. Bollen, H. Mao, and A. Pepe, “Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena.” in ICWSM, 2011, pp. 17–21.

[4] B. O’Connor, R. Balasubramanyan, B. R. Routledge, and N. A. Smith,“From tweets to polls: Linking text sentiment to public opinion time series.” in ICWSM, 2010, pp. 122–129.

[5] M. Hu and B. Liu, “Mining and summarizing customer reviews,” in KDD. ACM, 2004, pp. 168–177.

[6] T. Chen, R. Xu, Y. He, Y. Xia, and X. Wang, “Learning user and product distributed representations using a sequence model for sentiment analysis,” IEEE Computational Intelligence Magazine, vol. 11, no. 3, pp.34–44, 2016.

[7] Y. Wu, S. Liu, K. Yan, M. Liu, and F. Wu, “Opinionflow: Visual analysis of opinion diffusion on social media,” TVCG, vol. 20, no. 12, pp. 1763– 1772, 2014.

[8] E. Cambria, "Affective computing and sentiment analysis," IEEE Intelligent Systems, vol. 31, no. 2, pp. 102–107, 2016.

[9] E. Cambria, B. Schuller, Y. Xia, and B. White, "New avenues in knowledge bases for natural language processing," Knowledge-Based Systems, vol. 108, no. C, pp. 1–4, 2016.

[10] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: sentiment classification using machine learning techniques," in ACL, 2002, pp. 79–86.

[11] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision," CS224N Project Report, Stanford, pp. 1–12, 2009.

[12] F. Wu, Y. Song, and Y. Huang, "Microblog sentiment classification with contextual knowledge regularization," in AAAI, 2015, pp. 2332–2338.

[13] J. Blitzer, M. Dredze, F. Pereira et al., "Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification," in ACL, vol. 7, 2007, pp. 440–447.

[14] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in ICML, 2011, pp. 513–520.

[15] S.-S. Li, C.-R. Huang, and C.-Q. Zong, "Multi-domain sentiment classification with classifier combination," Journal of Computer Science and Technology, vol. 26, no. 1, pp. 25–33, 2011.

[16] L. Li, X. Jin, S. J. Pan, and J.-T. Sun, "Multi-domain active learning for text classification," in KDD. ACM, 2012, pp. 1086–1094.

[17] G. Li, S. C. Hoi, K. Chang, W. Liu, and R. Jain, "Collaborative online multitask learning," TKDE, vol. 26, no. 8, pp. 1866–1876, 2014.

[18] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," SIAM Journal on Imaging Sciences, vol. 2, no. 1, pp. 183–202, 2009.

[19] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," Foundations and Trends in Machine Learning, vol. 3, no. 1, pp. 1–122, 2011.

[20] F. Wu and Y. Huang, "Collaborative multi-domain sentiment classification," in ICDM. IEEE, 2015, pp. 459–468.

[21] S. Li and C. Zong, "Multi-domain sentiment classification," in ACL:HLT. Association for Computational Linguistics, 2008, pp. 257–260.

[22] S. J. Pan, X. Ni, J.-T. Sun, Q. Yang, and Z. Chen, "Cross-domain
sentiment classification via spectral feature alignment," in WWW. ACM,2010, pp. 751–760.

[23] Y. Lu, M. Castellanos, U. Dayal, and C. Zhai, "Automatic construction of a context-aware sentiment lexicon: an optimization approach," in WWW.ACM, 2011, pp. 347–356.

[24] S. J. Pan and Q. Yang, "A survey on transfer learning," TKDE, vol. 22, no. 10, pp. 1345–1359, 2010.

,