Recommending Ticket Resolution using Feature Adaptation

Wubai Zhou, Tao Li
School of Computing and Information Sciences
Florida International University
Miami, Florida 33199
Email: wzhou005, taoli@cs.fiu.edu

Larisa Shwartz
IBM T.J Watson Research Center
Yorktown Heights, New York, 10598
Email: lshwart@us.ibm.com

Genady Ya. Grabarnik
Department of Math and CS
St John's University
Queens, New York, 11439
Email: grabarng@stjohns.edu

***Abstract*—In recent years, IT Service Providers have been rapidly introducing automation to their service delivery model. Driven by market pressure to reduce cost and maintain quality of services, they are looking for technologies that will allow rapid progress towards attainment of truly automated service delivery.**

Software monitoring systems are designed to actively collect and signal event occurrences and, when necessary, automatically generate incident tickets. Repeating events generate similar tickets, which in turn have a vast number of repeated problem resolutions likely to be found in earlier tickets.

In our work, we develop techniques to recommend an appropriate resolution for incoming events by making use of similarities between the events and historical resolutions of similar events. The traditional KNN (K Nearest Neighbor) algorithm has been first applied to recommend resolutions for incoming tickets. Massive heterogeneous applications as well as various monitoring software are running on clients' servers to accomplish required tasks and to monitor system health via different metrics. It leads to generation of correlated tickets that have different symptom descriptions but similar resolutions. Furthermore, change of servers' environments can also induce similar situations in which ticket descriptions differ before and after change but could have similar resolutions. These correlated tickets cause performance degradation in ticket resolution recommendation. Therefore, we propose using SCL (structural corresponding learning) based feature adaptation to uncover feature mapping in different time intervals. Moreover, to put more insights into the periodic regularities existing in our ticket datasets, we apply our algorithm on tickets grouped by different time interval granularities. Extensive empirical evaluations on real-world ticket data sets demonstrate the effectiveness and efficiency of our proposed methods.

I. Introduction

The competitive business climate, as well as the complexity of service environments, dictate the need for efficient and cost-effective service delivery and support. These are largely achieved through service-providing facilities integrated with system management tools in combination with automation of routine maintenance procedures such as problem detection, determination and resolution for the service infrastructure [1], [2], [3], [4], [5]. Automatic problem detection is typically realized by system monitoring software, such as IBM Tivoli Monitoring [6] and HP OpenView [7]. Monitoring continuously captures the events and generates incident tickets when alerts are raised. Deployment of monitoring solutions is a first step towards fully automated delivery of a service. Automated problem resolution, however, is a hard problem.

However, most service providers keep years' worth of historical tickets with their resolutions. The resolution is usually collected as a free-form text and describes steps taken to remediate the issues in the ticket. We analyzed historical monitoring tickets collected from three different accounts managed by one of the large service providers (an account is an aggregate of services using common infrastructure). We noticed that there are many repeating resolutions for monitoring tickets within an account. It is natural to expect that if events are similar, then their respective tickets probably have the same resolution. Therefore, we can recommend a resolution for an incoming ticket based on the event information and historical tickets.

In our previous work [8], a KNN-based approach has been first applied to provide resolution recommendations for incoming tickets in service management. Additionally, several improved approaches [8], [9] have been proposed to resolve various shortcomings of the basic algorithm and thus to make recommended resolutions more relevant and practical. However, a further drawback has been uncovered when our previous methods were applied to system management.

In current service environments, massive heterogeneous applications, as well as various monitoring software, running on customers' servers to accomplish complex tasks and to monitor system health via different metrics, lead to generation of correlated tickets that have different symptom descriptions but similar resolutions. Furthermore, evolving over time, service environments cause a further discrepancy. The description of tickets generated before and after change differ but might have similar resolutions since root causes remain unchanged.

Based on our previous understanding and initial experiments, we find out that vocabularies used in ticket descriptions are changing and shifting over time but interesting mappings exist in those different vocabularies. However, our previous algorithms are not able to discover those mappings and thus their performance degrades over time due to inaccurate ticket similarity. To overcome drawback, we propose structural corresponding learning (SCL) to discover the words' mapping and apply it to our ticket resolution recommendation system.

The traditional KNN-based recommendation methodology was first proposed in our preliminary work [8], [9]. The details and extended algorithms are fully discussed there. The rest of our paper is organized as follows: Section II briefly introduces the workflow of the infrastructure management of an automated service and shares our observations on real-world monitoring tickets. In Section III, we present detailed implementation and application of SCL within resolution recommendation algorithms for monitoring tickets by finding feature mapping. Notice that phrases "feature mapping" and "feature adaptation" are used interchangeably in the rest of our work. In Section IV, we present experimental studies on real monitoring tickets. Section V describes related work about service infrastructure management, recommendation systems and SCL. Finally, Section VI concludes the paper and discusses our future work.

II. Background

In this section, we first provide an overview of automated service infrastructure monitoring with ticket generation and resolution. Then we present our analysis on real ticket data sets.

A. *Automated Services Infrastructure Monitoring and Event Tickets*

The typical workflow of problem detection, determination, and resolution in services infrastructure management is prescribed by the ITIL specification [10]. Problem detection is usually provided by monitoring software, which computes metrics for hardware and software performance at regular intervals. The metrics are then matched against acceptable thresholds. A violation induces an alert. If the violation persists beyond a specified period, the monitor emits an event. Events from the entire service infrastructure are accumulated in an enterprise console that uses rule-, case- or knowledge-based engines to analyze the monitoring events and decide whether to open an incident ticket in the ticketing system. The incident tickets created from the monitoring events are called monitoring tickets. Additional tickets are created upon customer request through Service Management System. The information accumulated in the ticket is used by technical support for problem determination and resolution. In this paper, we consider tickets generated by a monitoring system (see Figure 1).

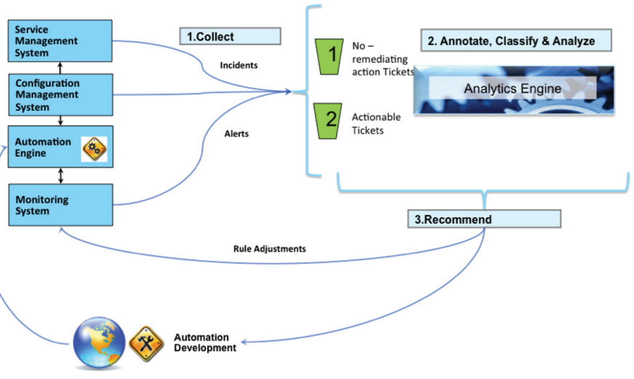

Fig. 1: Service Management System

Each monitoring ticket is stored as a database record that consists of several related attributes with values describing the system status at the time when monitoring event was generated. For example, a CPU-related ticket usually contains the CPU utilization and paging utilization information. A capacity-related ticket usually contains the disk name and the size of disk used/free space. Typically, different types of monitoring events have different sets of related attributes. The resolution of every ticket is stored as a textual description of steps taken by the system administrator to resolve this problem.

B. *Repeated Resolution of Monitoring Tickets*

We analyzed ticket data from three different accounts managed by IBM Global Services. Many ticket resolutions repeatedly appear in the ticket database. For example, for a low disk capacity ticket, usual resolutions mean deletion of temporal files, backup data, or addition of a new disk. Unusual resolutions are very rare.

TABLE I: Data Summary

Data set	Num. of Tickets	Time Frame
account1	31,447	1 month
account2	37,482	4 months
account3	29,057	5 months

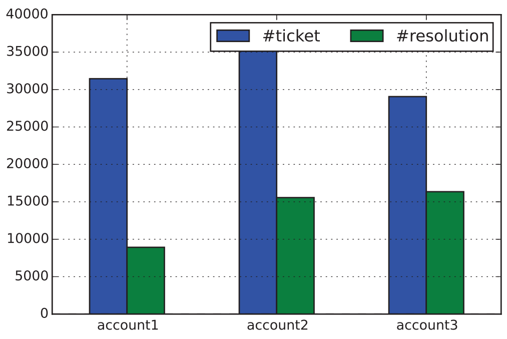

Fig. 2: Numbers of Tickets and Distinct Resolutions

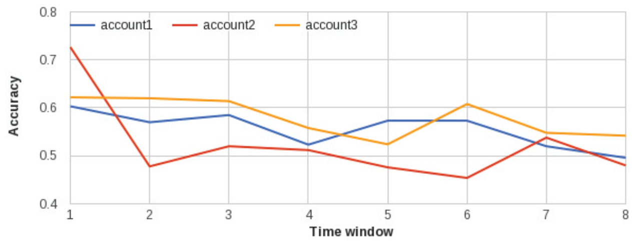

Fig. 3: Recommendation performance degrading as testing instances coming from different sliding window

Collected ticket sets from the three accounts are denoted by "account1," "account2" and "account3," respectively. Table I summarizes the three data sets. Figure 2 shows the numbers of tickets and distinct resolutions. We observe that a single resolution can resolve multiple monitoring tickets. In other words, multiple tickets share the same resolutions.

III. Feature Adaptation

In this chapter, we propose a solution for ticket resolution recommendation that accommodates vocabularies changing or shifting with time. We evaluate our solution on three real world ticket datasets collected from IBM service management system.

First, we show that feature variation and shift exists and degrades performance of ticket resolution recommendation. Second, we apply a structural correspondence learning (SCL) domain adaptation algorithm [11] for use in ticket resolution recommendation to solve the aforementioned issue. We assume that although features shift with time, there exists some feature mapping, i.e., some correspondence of features for tickets generated in different time intervals. We order and group tickets based on consecutive and disjointed time windows, and consider each time window as one domain. Thus, a feature mapping problem in ticket resolution recommendation become a domain or feature adaptation problem. We repeat process for each pair of consecutive sliding time windows.

In the following section we will briefly review SCL, introduce our new pivot selection strategy and describe how we apply SCL to ticket resolution recommendation problems.

A. Structural Corresponding Learning

First, we consider a simple example to illustrate application of SCL. Suppose that we have a dataset of historical tickets based on which we need to identify an appropriate resolution for the incoming event. Resolutions for the same root cause could slightly differ, but descriptions of symptoms could vary significantly. For example, two tickets have the same resolution as ”archive the logs and thus reduce the space utilization”, but their descriptions could be different when diverse vocabularies are used, such as ”volume”, ”capacity” or ”harddiskvolume”.

Our key intuition is that even if ”volume”, ”capacity” and ”harddiskvolume” are literally distinct, they have high correlation with ”space” or ”utilization” in the historical data set, and thus we can tentatively construct some mapping between them and recommend similar resolutions for incoming events represented by different vocabulary.

B. Algorithm Overview

Given two consecutive sliding time windows, source tickets are defined as the tickets from the first time window; target domain tickets are the tickets from the second time window. The SCL first chooses a set of m pivot features that occur frequently in both domains. Next, it models the correlation between the pivot features and all other features by training pivot predictors to predict the occurrences of each pivot feature in all training dataset from both domains [12], [11]. The coefficients of the l-th pivot predictor characterize the correlation between non-pivot features with the l-th pivot feature; positive coefficients indicate that a non-pivot features is highly correlated with the corresponding pivot feature.

We consider the coefficients of each pivot predictor as a column vector. All predictors can then be arranged into a matrix $W = [w_l]_{l=1}^{n}$, where w_l is the lth column coefficient vector and n is the number of pivot features. Let $\theta \in \mathbb{R}^{h \times d}$ be the top h left singular vectors of W, i.e.,

$$[U\ D\ V^T] = SVD(W),\ \theta = U^T[1:h,:]. \qquad (1)$$

There vectors are the principal predictors for our coefficient space. If these pivot features are well chosen, we expect these principal predictors to distinguish between words leading to similar and different resolutions in both domains.

As we observe a feature vector $x \in S$ at training and testing time, we notice feature space S is different for different domains. We apply the projection θx to obtain k new real-valued features. Now we use augmented feature vector $< x, \theta x >$ for the same instance. If θ contains meaningful correspondences, then we have a mapping of feature vectors from different feature spaces into a shared feature space. The shared features will bring tickets predicting similar pivot features closer using similarity measurement given in following Equation 2:

$$sim(xa_s, xa_t) = \frac{\sum_{w \in V} x_s(w) * x_t(w)}{2|x_s| \cdot |x_t|} + \frac{cos(\theta x_s, \theta x_t)}{2} \qquad (2)$$

Where $xa =< x, \theta x >$ is the augmented feature vector, V is the shared words in two feature space that x_s and x_t belongs to, $x(w)$ is the entry value for word w in vector x and $cos(\cdot,\cdot)$ is the cosine similarity function. Here we assume those pivot features strongly indicate the resolution and we will explain how we extract those pivot features in the next section.

C. Picking Pivot Features

The efficacy of SCL depends on the choice of pivot features. In [11], frequently-occurring words are chosen as pivot features to resolve domain adaptation in a speech tagging problem. Frequently-occurring words often correspond to function words, such as prepositions and determiners, and are good indicators of parts of speech. With respect to sentiment classification in [13], those features are chosen as pivot features which have the highest mutual information to the source label. We use a different approach, however, for a ticket resolution recommendation task in picking up pivot features. We require pivot features to be good predictors of resolutions. We attempt following two approaches in our situations.

First, we calculate the term frequency-inverse document frequency (TF-IDF [14]) scores for all words out of *ticket symptom description* in both domains and choose 1000 words having the highest TF-IDF scores for each domain. Then, we choose the m most frequently-occurring words out of the two sets of 1000 words. This allows us to eliminate function words, such as prepositions and determiners, while choosing less frequently-occurring words that are strong indicators of resolutions. However, vocabularies used for describing ”ticket symptom” and ”ticket resolution” could differ, and the first approach only gives us the words that are strong indicators of ”symptom” instead of ”resolution”. Second, we assume that there are some tickets with resolution in target domain and use the strategy shown in Table III to pick the pivot features from *ticket resolutions* instead of ticket symptoms. The assumption can be easily satisfied in a practical scenario since we define source and target domain via partitioning the tickets and, therefore, we can always assign those latest tickets with resolutions to a target domain. Table II contains the top pivot features chosen using these two approaches.

TABLE II: Top pivot features chosen from description and resolution

DESCRIPTION	RESOLUTION
app space job high restore status error procedures failed db	incident close copy resolve server found issue action team job clear close file

As shown in Table II, the pivot features chosen from description strongly describe the ticket symptom observed on the server system. At the same time, the pivot features chosen by resolution describe the ticket resolution, i.e., how to resolve issues on the server system.

D. Pivot Predictors

From each pivot feature we create a binary classification problem of the form "does pivot feature l occur in this ticket?". Then we classify the training set. If we represent our features as a vector x, we can solve these problems using m linear predictors in which we use a linear regression model with l_2 regularization as the underlying classification model.

$$f_l(x) = sgn(\hat{w}_l \cdot x), l = 1 \dots m \quad (3)$$

Table III summarizes the details about construction of predictors given by Equation 3.

TABLE III: Details on predictors' construction

$sgn(\cdot)$	does pivot feature l occurs in the resolution of this ticket?
pivot features	the m most frequently-occurring words shared in the two sets of 1000 words having the top TF-IDF scores in ticket resolutions of both domains
x	1000 words having the highest TF-IDF scores from symptom description
training data	all tickets attached with resolution from both domains

The pivot predictors are the key element in SCL. The weight vectors $\hat{w}_l$ encode the covariance of the non-pivot features with the pivot features. If the weight given to the z-th feature by the l-th pivot predictor is positive, then feature z is positively correlated with pivot feature l. Since pivot features strongly indicate resolutions, we expect non-pivot features from both domains to be correlated with them. If two non-pivot features are correlated in the same way with many of the same pivot features, then they have a high degree of correspondence.

Our algorithm has two hyper parameters, i.e., the number of pivot features m and the h in Equation 1. We set $h = 30$ and $m = 70$ according to prior work and experiments in our work which are not shown here due to limited contents.

IV. Evaluation

In this chapter, we will focus on the dataset, the running environment and discussion of experimental results.

A. Setup

For each account, we ordered tickets by time and chose various approaches to slide the dataset. Codes are implemented in Java, running on 64-bit Windows 7 operating system residing on a machine equipped with 16 GB RAM, Intel(R) Core(TM) i7-4770 CPU running at 3.40 GHz. Training of pivot predictors is parallelized.

B. Evaluation of Feature Adaptation

The goal of our first experiment is to verify feature adaptation in which tickets are ordered by time. The first 6000 tickets are chosen for experiments. Then it has been consecutively partitioned into three parts. The first two are considered as source and target tickets for training and the third part as testing pool from which we sample 400 testing instances. Pivot features is extracted from available resolutions of both source and target domain using the approach shown in Table III . Under this setup, our experiments show that feature mapping between the vocabularies used in both source domain and target domain exists, and it strongly helps in improving ticket recommendation performance.

Figure 4 shows the overall performances using two algorithms "No-TF" and "TF-3". "No-TF" is just the basic KNN-based recommendation algorithm with no feature adaptation and "TF-3" is the one with feature adaptation. As shown

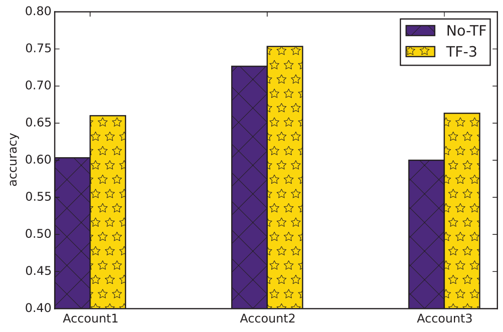

Fig. 4: Overall performance for three accounts

in Figure 4, "TF-3" outperforms "No-TF" by average 9%, 3%, 10% respectively for three accounts. We select an event ticket in account1 to illustrate why "TF-3" is better than the basic KNN-based algorithm "No-TF." Table IV shows a list of recommended resolutions given by each algorithm. The testing ticket is a real event ticket triggered by an error when processing a text file. It has the symptom description as "an error in process xxx while processing file xxx.txt, leave the processing" and its true resolution is "connectivity issue, the file has been retransmitted successfully." The general idea of this resolution is to retransfer and reprocess the file.

As shown in Table IV, the second resolution recommended by "No-TF" is the most relevant but still a wrong resolution: "as a part of application team testing, file has been successfully repulled." Obviously, it is caused by manual testing. "TF-3," however, recommends 4 resolutions all highly relevant to the true resolution. According to the definition of *HIT* [8], the first resolution recommended by "TF-3" is regarded as a true resolution, i.e., "the file has been retransmitted successfully." Also as we notice, the word "file" is a pivot feature shown in Table III that allow us to identify expected resolutions. Therefore, our proposed algorithm "TF-3" indeed can find the hidden feature mapping using SCL and the feature mapping performs better for recommending ticket resolution.

TABLE IV: Case Study

algorithm	recommended resolutions	isHit
No-TF	closing this ticket as its a duplicate of the incident inc0771310	FALSE
	as a part of application team testing, file has been successfully repulled	FALSE
	the file is been decommissioned, no action requried, hence resolving	FALSE
TF-3	the file has been retransmitted successfully	TRUE
	the file was delivered successfully	FALSE
	the file has been successfully repulled	FALSE
	file was pulled successfully from bank of xxx	FALSE

Moreover, we visualized one row of the project matrix θ for our experiments on general feature adaptation. Table V illustrates the first row of θ; the features on each row appear only in the corresponding domain. In a traditional binary classification problem of applying SCL [13], corresponding features indicate either a positive or negative label. Corresponding features here indicate event tickets caused by the same or similar root cause and thus share similar resolutions. We colored those correspondence feature groups so they could be easily identified visually. For example, features "*sdump, page, harddiskvolume, paging, traps*" colored in red indicate system issues in or similar to paging due to low capacity. "sdump" is an excutable command that tries to dump virtual storage and thus makes space for paging. Without feature mapping, event tickets will be considered having low or no similarity if they contain discriminant features. Once we discovered feature mapping, we can project those discriminant features to shared feature space by applying them to θx. The features will ensure that their corresponding event tickets have higher similarity.

type	features
+s	sdump bee idc ami sr included refer read processing queues page read-response
+t	readresponse getacctsummbycustid contingency cli logerror harddiskvolume paging traps getacctsforgrantee ant
-s	messages wtprocess normal wiptrigger ifscmonitor poa responding dump-code acctinfo
-t	batjbstrm sm fndstn xmx dbm aelv throw responsestream

TABLE V: Correspondences discovered by SCL for general feature mapping experiments. Notation "s" corresponds to features coming from source domain, and "t" corresponds to features coming from target domain. The "+" and the "-" symbols indicate positive and negative features in correspondences, respectively.

C. *Feature Adaptation for Different Time Granularity*

In section IV-B, we discussed the experiments for general feature adaptation problems. In this section we consider an adaptability between ticket data sets sliced by different time granularities, i.e., daily, weekly or even monthly. The goal of this experiment is to understand the feature shifting phenomenon and the shifting of an event type in different time granularities. Positive results in recommendation task would indicate that monitoring tickets generated daily do not change too much and there exists stable pivot features for constructing meaningful feature mapping in the feature vocabulary. Otherwise, it would indicate that the daily monitoring tickets shift a lot leading to various daily resolutions and thus noisy pivot features.

Figure 5 shows the experimental results for daily feature adaptation. Tickets from three consecutive days are required for one trial. "No-TF" only uses first day's data for training and last day's for testing, meanwhile "TF-3" uses the first two days' data as source and target domain respectively for training and last day's for testing. The next trial is based on the tickets from a time window that we get by sliding the start date of time window one time unit later. Weekly consecutive feature adaptation experiments have been carried out in a similar setup and the results are shown in Figure 6 and Figure 7 for account1 and account2 respectively. Account3 has been ruled out of weekly feature adaptation experiments since it only has two weeks tickets, and the experiment requires ticket data from at least three weeks.

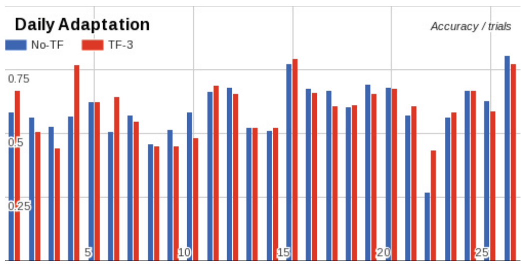

Fig. 5: Daily adaptation for account1.

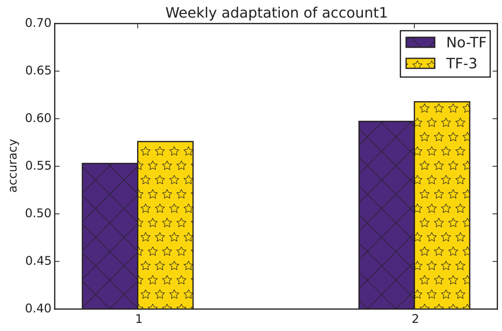

Fig. 6: Weekly adaptation experimental results on account1's four weeks data. Two trials are carried out since each trial requires three weeks data.

While feature mapping learned from the first two consecutive days' tickets are useful for recommending the last day's event ticket resolutions from the first day, it can also degrade the recommendation performance. This causes problems when resolutions indicated by pivot features are quite commonly shared for the first two days but not for the third day. For example, the event tickets occurred in the first two days mainly caused by "software exception" and "system failure" but in the last day they occurred because of "low capacity." As

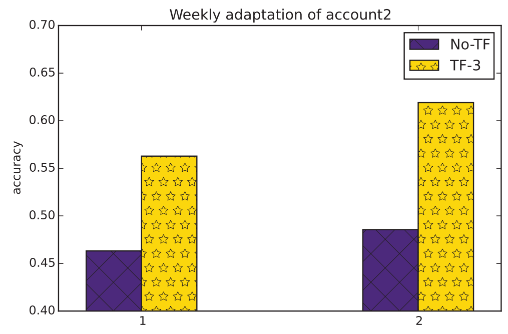

Fig. 7: Weekly adaptation experimental results on account2's four weeks data. Two trials are carried out since each trial requires three weeks data.

shown in Figure 5, around half of the trials show performance degradation for our approach "TF-3." These feature mappings, which are not applicable between the first daily tickets and the last daily ones, are used to project features of the last daily tickets into shared feature space with first daily tickets. They cause noisy and inaccurate similarity calculations by recommending algorithm and degradation of accuracy in resolution recommendation .

Our experiment on weekly ticket datasets achieved positive results as illustrated in Figure 6 and Figure 7. They indicate that the distribution of event types occurring weekly are similar for these three consecutive weeks.

V. Related Work

This section reviews prior research studies related to the automated IT service management and domain adaptation. System monitoring, as part of the automated service management, has become a significant research area of the IT industry in the past few years. Commercial products, such as IBM Tivoli [15], HP OpenView [7] and Splunk, [16] provide system monitoring. Numerous studies [17], [18], [19], [20], [21], [22] focus on monitoring that is critical for a distributed network. The monitoring targets include the components or subsystems of IT infrastructures, such as the hardware of the system (CPU, hard disk) or the software (a database engine, a web server). Once certain system alarms are captured, the system monitoring software will generate the event tickets into the ticketing system. Automated ticket resolution is much harder than automated system monitoring because it requires vast domain knowledge about the target infrastructure. Some prior studies apply approaches in text mining to explore the related ticket resolutions from the ticketing database [23], [24]. Other works propose methods for refining the work order of resolving tickets [23], [25]. A number of studies focused on the analysis of time series event data and textual log files with the goal of improving an understanding of system behaviors [26], [27], [28], [29]. Another area of interest is the identification of actionable patterns of events and misses, or false negatives, by the monitoring system [30]. False negatives are indications of a problem in the monitoring software configuration, wherein a faulty state of the system does not cause monitoring alerts.

Large service providers staff their service centers with hundreds of IT experts who are responsible for resolving various incident tickets every day. Therefore, service providers heavily rely on human efficiency for tasks such as root cause analysis and incident ticket resolution. Automatic techniques of recommending relevant historical tickets with resolutions can significantly improve the efficiency of technical support in this task. In our work, we resort t domain adaptation techniques SCL to further improve our previous work [9] in ticket resolution recommendation.

Domain adaptation is a well studied area. Roark and Bacchiani [31] use a Dirichlet prior on the multinomial parameters of a generative parsing model to combine a large amount of training data from a source corpus and a small amount of training data from a target corpus. Several authors have also given techniques for adapting classification to new domains. Chelba et al. [32] first train a classifier on the source data and then apply the maximum a posteriori estimation of the weights of a maximum entropy on a target domain classifier in which the Gaussian prior has a mean equal to the weights of the source domain classifier. Daumé III and Marcu [33] use an empirical Bayes model to estimate a latent variable model grouping instances into domain-specific or common across both domains. Our work focuses on applying SCL to find a common representation for features from different tickets to favor ticket resolution recommendation.

Finally we note that SCL is first introduced in the work of Ando et al. [12], and later Blitzer combines SCL with labeled target domain data, they compared the two using the label of SCL or non-SCL source classifiers as features. Several applications of SCL have been studied in papers [11], [13]. Unlike these applications, we apply SCL to our ticket resolution recommendation task and pick up the pivot features from both source and target labeled tickets. We show that we can make better use of SCL to discover a useful feature mapping in our real-work ticket data and improve performance of our ticket resolution recommendation task.

VI. Conclusion

This paper studies the problem of resolution recommendation for monitoring tickets in an automated service management. Based on our previous work and some initial experiments, we observe the feature shifting phenomon and the existence of feature mapping in those tickets. In this paper, we applied structural correspondence learning to the problem of recommending ticket resolution, and conducted extensive experiments on real-world ticket data sets to demonstrate the effectiveness and efficiency of proposed methods.

There are several avenues for future research. First, we plan to investigate and develop intelligent classification techniques to automatically label resolutions [34], [35]. Second, our current recommendation system uses KNN-based algorithms due to their simplicity and efficiency. We will investigate and develop other advanced algorithms to improve the recommendation performance. Finally, we also plan to use an active query strategy to fully automate resolution recommendations.

Acknowledgment

This work is partially supported by the U.S. National Science Foundation under grants IIS-0546280, HRD-0833093, and IIS-1213026.

References

[1] P. Marcu, L. Shwartz, G. Grabarnik, and D. Loewenstern, "Managing faults in the service delivery process of service provider coalitions," in *IEEE SCC*, 2009, pp. 65–72.

[2] L. Tang, T. Li, F. Pinel, L. Shwartz, and G. Grabarnik, "Optimizing system monitoring configurations for non-actionable alerts," in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, 2012.

[3] N. Ayachitula, M. J. Buco, Y. Diao, M. Surendra, R. Pavuluri, L. Shwartz, and C. Ward, "IT service management automation - a hybrid methodology to integrate and orchestrate collaborative human centric and automation centric workflows," in *IEEE SCC*, 2007, pp. 574–581.

[4] B. Wassermann and W. Emmerich, "Monere: Monitoring of service compositions for failure diagnosis," in *ICSOC*, 2011, pp. 344–358.

[5] Y. Yan, P. Poizat, and L. Zhao, "Repair vs. recomposition for broken service compositions," in *ICSOC*, 2010, pp. 152–166.

[6] "IBM Tivoli Monitoring," http://ibm.com/software/tivoli/products/monitor/.

[7] "HP OpenView : Network and Systems Management Products," http://www8.hp.com/us/en/software/enterprise-software.html.

[8] L. Tang, T. Li, L. Shwartz, and G. Grabarnik, "Recommending resolutions for problems identified by monitoring," in *Integrated Network Management (IM 2013), 2013 IFIP/IEEE*. IEEE, 2013, pp. 134–142.

[9] W. Zhou, L. Tang, T. Li, L. Shwartz, and G. Y. Grabarnik, "Resolution recommendation for event tickets in service management," in *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*. IEEE, 2015, pp. 287–295.

[10] "ITIL," http://www.itil-officialsite.com/home/home.aspx.

[11] J. Blitzer, R. McDonald, and F. Pereira, "Domain adaptation with structural correspondence learning," in *Proceedings of the 2006 conference on empirical methods in natural language processing*. Association for Computational Linguistics, 2006, pp. 120–128.

[12] R. K. Ando and T. Zhang, "A framework for learning predictive structures from multiple tasks and unlabeled data," *The Journal of Machine Learning Research*, vol. 6, pp. 1817–1853, 2005.

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

[14] C. D. Manning, P. Raghavan, H. Schütze *et al.*, *Introduction to information retrieval*. Cambridge university press Cambridge, 2008, vol. 1.

[15] "IBM Tivoli : Integrated Service Management," http://ibm.com/software/tivoli/.

[16] "Splunk: A commerical machine data managment engine," http://www.splunk.com/.

[17] S. R. Kashyap, J. Ramamirtham, R. Rastogi, and P. Shukla, "Efficient constraint monitoring using adaptive thresholds," in *Proceedings of ICDE*, Cancun, Mexico, 2008, pp. 526–535.

[18] S. Agrawal, S. Deb, K. V. M. Naidu, and R. Rastogi, "Efficient detection of distributed constraint violations," in *Proceedings of ICDE*, Istanbul, Turkey, 2007, pp. 1320–1324.

[19] S. Mccanne and V. Jacobson, "The bsd packet filter: A new architecture for user-level packet capture," in *USENIX Technical Conference*, 1993, pp. 259–270.

[20] K. Xu, Z.-L. Zhang, and S. Bhattacharyya, "Profiling internet backbone traffic: behavior models and applications," in *ACM SIGCOMM Conference*, 2005, pp. 169–180.

[21] C. Estan, S. Savage, and G. Varghese, "Automatically inferring patterns of resource consumption in network traffic," in *ACM SIGCOMM Conference*, 2003, pp. 137–148.

[22] M. J. Ranum, K. Landfield, M. T. Stolarchuk, M. Sienkiewicz, A. Lambeth, and E. Wall, "Implementing a generalized tool for network monitoring," in *USENIX SysAdmin Conference*, 1997, pp. 1–8.

[23] Q. Shao, Y. Chen, S. Tao, X. Yan, and N. Anerousis, "EasyTicket: a ticket routing recommendation engine for enterprise problem resolution," *PVLDB*, vol. 1, no. 2, pp. 1436–1439, 2008.

[24] D. Wang, T. Li, S. Zhu, and Y. Gong, "iHelp: An intelligent online helpdesk system," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 41, no. 1, pp. 173–182, 2011.

[25] G. Miao, L. E. Moser, X. Yan, S. Tao, Y. Chen, and N. Anerousis, "Generative models for ticket resolution in expert networks," in *KDD*, 2010, pp. 733–742.

[26] C. Zeng, L. Tang, T. Li, L. Shwartz, and G. Y. Grabarnik, "Mining temporal lag from fluctuating events for correlation and root cause analysis," in *Network and Service Management (CNSM), 2014 10th International Conference on*. IEEE, 2014, pp. 19–27.

[27] J. L. Hellerstein, S. Ma, and C.-S. Perng, "Discovering actionable patterns in event data," *IBM Systems Journal*, vol. 43, no. 3, pp. 475–493, 2002.

[28] C. Perng, D. Thoenen, G. Grabarnik, S. Ma, and J. Hellerstein, "Data-driven validation, completion and construction of event relationship networks," in *Proceedings of the ninth ACM SIGKDD*. ACM, 2003, pp. 729–734.

[29] G. Grabarnik, A. Salahshour, B. Subramanian, and S. Ma, "Generic adapter logging toolkit," in *Autonomic Computing, 2004. Proceedings. ICAC*. IEEE, 2004, pp. 308–309.

[30] L. Tang, T. Li, L. Shwartz, and G. Y. Grabarnik, "Identifying missed monitoring alerts based on unstructured incident tickets," in *Network and Service Management (CNSM), 2013 9th International Conference on*. IEEE, 2013, pp. 143–146.

[31] B. Roark and M. Bacchiani, "Supervised and unsupervised pcfg adaptation to novel domains," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. Association for Computational Linguistics, 2003, pp. 126–133.

[32] C. Chelba and A. Acero, "Adaptation of maximum entropy capitalizer: Little data can help a lot," *Computer Speech & Language*, vol. 20, no. 4, pp. 382–399, 2006.

[33] H. Daume III and D. Marcu, "Domain adaptation for statistical classifiers," *Journal of Artificial Intelligence Research*, pp. 101–126, 2006.

[34] C. Zeng, T. Li, L. Shwartz, and G. Y. Grabarnik, "Hierarchical multi-label classification over ticket data using contextual loss," in *Proceedings of IEEE/IFIP NOMS*. IEEE, 2014, pp. 1–8.

[35] S. Chang, G.-J. Qi, J. Tang, Q. Tian, Y. Rui, and T. S. Huang, "Multimedia lego: Learning structured model by probabilistic logic ontology tree," in *Data Mining (ICDM)*. IEEE, 2013, pp. 979–984.