

Multi-focal Learning and Its Application to Customer Service Support

Yong Ge¹, Hui Xiong¹, Wenjun Zhou¹, Ramendra Sahoo², Xiaofeng Gao³, Weili Wu³

¹ MSIS Department, Rutgers Business School, Rutgers University
hxiong@rutgers.edu, {yongge, wjzhou}@pegasus.rutgers.edu

² IBM T.J. Watson Research Center, rsahoo@us.ibm.com

³ Department of Computer Science, University of Texas - Dallas

ABSTRACT

In this study, we formalize a multi-focal learning problem, where training data are partitioned into several different focal groups and the prediction model will be learned within each focal group. The multi-focal learning problem is motivated by numerous real-world learning applications. For instance, for the same type of problems encountered in a customer service center, the problem descriptions from different customers can be quite different. The experienced customers usually give more precise and focused descriptions about the problem. In contrast, the inexperienced customers usually provide more diverse descriptions. In this case, the examples from the same class in the training data can be naturally in different focal groups. As a result, it is necessary to identify those natural focal groups and exploit them for learning at different focuses. The key developmental challenge is how to identify those focal groups in the training data. As a case study, we exploit multi-focal learning for profiling problems in customer service centers. The results show that multi-focal learning can significantly boost the learning accuracies of existing learning algorithms, such as Support Vector Machines (SVMs), for classifying customer problems.

Categories and Subject Descriptors

H.2.8 [Database Management]: Database Applications—*Data Mining*; I.5.2 [Pattern Recognition]: Design Methodology—*Classifier Design and Evaluation*

General Terms

Algorithms, Experimentation

Keywords

Multi-focal Learning, Customer Service Support

1. INTRODUCTION

Customer service support is becoming an integral part of most companies. Many companies have a customer service

department that provides inspection, installation, and maintenance support for their world-wide customers. Problem ticket generation usually is the first step in today's process services management. This step is responsible for describing problem symptoms reported by customers and problem tickets are the link between customers and the services infrastructure. Once a problem ticket is generated, it will be enqueued in the ticketing system and routed to an appropriate service center or service people for problem determination. Most existing large call centers collect service data that are then used to assess and improve the performances of their representatives. Typically, these problem records are stored in relational databases with both structured (e.g. current status, problem type, support person/group handling the problem, type of system and component related to the problem) as well as unstructured (e.g. free-format text descriptions of problems and solutions as entered by the support personnel) attributes.

Indeed, the increasing availability of problem logs creates unprecedented opportunities to change the paradigm for risk management for avoiding/alleviating organizational crisis, product design, and product quality control. Currently, these problem logs are mostly used for tracking, auditing and reporting the problem management processes [4, 11, 3]. Most steps in these processes (e.g., problem diagnosis, ticket routing, etc.) are still manually taken. However, a lot of subject knowledge and experiences that these manual steps rely on are embedded in the existing problem records (i.e. historical data) [14, 16]. It is expected that we develop the ability to automatically extract the expert experiences as the knowledge to improve process services management and identify early symptoms of defect products.

To this end, in this study, we aim to exploit large-scale problem logs for predicting problem category/determination automatically. This is a main aspect of customer service profiling [8]. If problems can be automatically determined, the problems can be routed to the right support group/personnel more efficiently. In turn, this can offer an organization tremendous benefit by, for instance, identifying sources of problem resolution error, delay, and optimization of the problem management processes. Also, this can help to reduce the expense caused by manual processing. Furthermore, it is possible to facilitate customer service profiling if we have the ability to automatically categorize problems.

While it is appealing to automate the process of problem categorization/determination, it is very challenging to make use of the information and expert knowledge encoded in problem logs. For problems reported by customers, the best

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

KDD'09, June 28-July 1, 2009, Paris, France.

Copyright 2009 ACM 978-1-60558-495-9/09/06 ...\$5.00.

Table 1: A Sample Problem Log Entry

ProblemID	CauseCode	System	OccurredDate/Time	Component	ReportGroup	ResolverGroup
30446586	APPLICATION	E-ESM	01-Jan-07/00.18.55	PROBLEM	AINTT HDL2PW	AINTT HDL2PW
Problem Description				Problem Solution		
<i>I changed my password over the holidays and I didn't write it down. Now I can't remember it.</i>				<i>Left a message to get the ID file and password and closing the ticket</i>		

information we can use is the problem description provided by customers. If we treat the problem descriptions as training samples and the problem result categories (determined by experts) as labels, it appears that existing off-the-shelf classification algorithms [17] can be used for problem categorization. However, after carefully examining real-world problem logs, we have observed some unique characteristics inherent in problem logs. A key issue is that all the problem descriptions for the same problem are provided by customers with diverse background and these problem descriptions can be quite different. The experienced customers usually give more precise and focused descriptions about the problem. In contrast, the inexperienced customers usually provide diverse descriptions for the same problem. As an example, for a simple password problem, the description from experienced customers can be as simple as “need to reset password”. However, inexperienced customers provide much more diverse descriptions, such as “cannot connect to the network” and “my computer does not work”. In other words, the training samples for the same class can be naturally in different focal groups. If we treat these problem descriptions as the same and fit into existing learning models, we may not be able to have desirable classification performances.

The above observations cast the light on the major body of this research. Specifically, we formalize a multi-focal learning problem, where training data are partitioned into several different focal groups and the prediction model will be learned within each focal group. A key developmental challenge is how to identify those focal groups in the training data. As mentioned above, for the same problem, the problem descriptions from experienced customers are very precise and focused. Accordingly, the descriptions of problem solutions are also very precise and focused and highly correlated with the problem descriptions by experienced customers (if descriptions have been turned into vectors of words). In contrast, the problem descriptions for the same problem from inexperienced customers are weakly correlated with the descriptions of their corresponding problem solutions. Viewed in this light, we propose a correlation method (CORRELATION) to partition problem descriptions within each class into two different groups: one for experienced customers and the other for inexperienced customer. In addition, to better capture the information encoded in problem logs, we also develop an ontology-enhanced correlation method (ONTOLOGY) for identifying different focal groups. After the learning models are constructed for different focal groups, the new samples will be assigned to a learning model based on a nearest neighbor method; that is, a new sample will be assigned to a learning model if this sample is closer to the centroid of the train samples for this learning model than that of any other model.

To evaluate the performances of multi-focal learning, we present a theoretical risk analysis of multi-focal learning with the Naïve Bayes classifier and reveal that the risk of multi-focal learning with the Naïve Bayes classifier is smaller than the risk of the single Naïve Bayes learning model. Moreover, we exploit the multi-focal learning model for categorizing

problems using real-world problem logs. The experimental results show that multi-focal learning can significantly boost the learning accuracies of existing learning algorithms, such as SVMs and RIPPER [18]. Finally, we show that both CORRELATION and ONTOLOGY can lead to a better learning performance than other focal-group formation methods, such as the methods based on clustering and random-partition. However, ONTOLOGY results in slightly better learning performances than CORRELATION.

2. OVERVIEW OF PROBLEM LOGS

We aim to develop a problem categorization model for facilitating customer service analysis. To achieve this, we build a prediction model based on the problem logs collected from customer service centers. While the problem logs used here are only related to IT Enterprise products and service functions, the developed prediction methodology and the prediction framework can be easily extended to deal with problem logs from broader business sectors.

Problem logs used in this paper were collected from IBM customer service centers. These problem logs are mainly IBM customer service texts describing the customer problems and these texts were recorded during problem incidences. While these problem logs record issues dominated by IBM related products and services, there are some problem logs which report problems from other vendors as well. All problem logs are stored in relational databases with both structured (e.g. problem ID, cause code, Person/group handling the problem, type of system and component related to the problem) as well as unstructured (e.g. free-text descriptions of problem and problem solution descriptions entered by the support personnel) attributes. Table 1 shows a sample problem log. In the table, we can observe some structured attributes as well as two unstructured attributes of this sample problem log. As can be seen, problem descriptions are about the descriptions of problems and were supplied by customers with diverse background. In contrast, the descriptions about problem solution were provided by IBM service people. Both problem descriptions and their problem solutions are in the free-text format.

In this study, we focus on the unstructured attributes, since the free-text descriptions of problems supplied by customers are original and essential for the solutions of the problems. Compared with structured data, unstructured free-text descriptions offer more original information that will reveal “*what*” and “*why*” aspects of the problems that a customer might have encountered. However, the analysis of unstructured text is extremely difficult because of the following reasons. First of all, any unstructured text is usually very noisy with many irrelevant text contents [6]. Second, the problem descriptions usually reflect the problem perceptions of customers and are given in customers’ own words. In other words, the same problem can be described in many different ways since the different customers have very different levels of domain knowledge.

Finally, before we can apply learning models on problem logs for predicting problem categorization, we need to

first do data preprocessing, which includes data cleaning and data transformation. For data transformation, we transform unstructured problem descriptions and problem solutions into structured vectors. This involves extracting keywords from text paragraphs and then representing each problem log as a vector with 0/1 values. This data pre-processing is similar to that in text categorization. Note that we have used the Porter word stemming algorithm [13] to reduce words to their base stem.

3. MULTI-FOCAL LEARNING

In this section, we first introduce the concept of multi-focal learning. Then, we explain how the multi-focal learning model works by providing an illustrating example. In addition, we propose two methods for forming focal groups in problem logs. Finally, we provide a theoretic analysis to show the effectiveness of the multi-focal learning.

3.1 An Overview of Multi-focal Learning

The idea of multi-focal learning is motivated by the observation that there are inherent large variations in many real-world training data. For instance, problem descriptions in problem logs for the same problem can be quite different, since these descriptions may be from customers with different background. Some customers may be experienced people and they can provide more precise descriptions about the problems. In contrast, some inexperienced customers may provide diverse descriptions on the same problem. As a result, the learning performances can be significantly impacted by the diversity inherent in the training data. This is referred as the multi-focal property in this paper.

To deal with this multi-focal property, we provide a multi-focal learning framework as shown in Figure 1. As can be seen, there are three phases for the multi-focal learning. In the first and the most challenging phase, there is a need to identify different focal groups in a way such that training samples in the same focal group are more similar to each other. This focal-group formation process is challenging because there is no simple and effective way to identify focal groups in different types of real-world data. The best focal-group formation method usually relies on the special characteristics inherent in the data. Here, we provide two methods for finding focal groups in problem logs in the following subsections. The second phase is focused on building learning models in each focal group. Any traditional learning models, such as Support Vector Machines (SVMs), Bayesian Learning, and Decision Trees, can be applied here. Finally, in the third phase, the test samples will be first assigned to a focal group using a nearest neighbor method. Then, the learning model from the corresponding focal group will be used for the learning on this test sample.

3.2 An Illustration of Multi-focal Learning

Here, we exploit SVMs on a synthetic data set to illustrate multi-focal learning. Specifically, we generate two-dimensional synthetic data with two classes as shown in Figure 2 (a). In the figure, the objects of two classes are represented by triangle and square symbols respectively and these objects are naturally located in two different focal groups: one dense group and one sparse group. Note that the SVMs tool we used here is LIBSVM [2] with the linear kernel.

First of all, we generate the learning model by directly applying SVMs for the whole original data and the results are

Multi-focal Learning

Input: TR: a training data set.
 TE: a test data set.
 LCF: a classifier, such as SVMs.
 K: the number of focal groups in training data.
Note K=2 in this study.
 Output: CMs: the multi-focal models built on TR.
 CR: the prediction results.

Procedure:

Phase I: Identifying focal Groups

1. if do local partition within each class
2. for class $i=1$ to $C // C: \#classes$
3. $Group_{(k=1, \dots, K)}^i = Partition(TR(i), K);$
4. end for
5. for group $k=1$ to K
6. $Group_k = merge_{i=1, \dots, C} Group_k^i$
7. end for
8. else do global partition in all classes
9. $Group_{(k=1, \dots, K)} = partition(TR, K)$
10. end if

Phase II: Training the model in each focal group.

11. for group $k=1$ to K
12. $CM(k) = train(Group_k, LCF);$
13. end for

Phase III: Testing/prediction

14. $CM^* = choose(Groups, TE);$
15. $predictLabel = predict(CM^*, TE)$

Figure 1: The Multi-focal Learning Framework

show in Figure 2 (b). In this figure, the solid line represents the maximal margin hyperplane (MMH) learned by SVMs. As can be seen, there are many classification errors. This is due to the multi-focal property of the original data. Indeed, even if we use non-linear kernels (i.e. the Radial Basis Function), the classification accuracy is still about 60%.

Instead, we exploit multi-focal learning methods. Along this line, we first partition the data into two groups. One is the dense group and the other is the sparse group as shown in Figure 2 (c). In this figure, we can see that samples from two classes co-exist in these two focal groups. Next, we apply SVMs to build learning models in each group and the results are shown in Figure 2 (d). As can be seen, there is one MMH in each group and there are few classification errors compared to Figure 2 (b). Finally, to predict a test sample in the multi-focal learning framework, we first assign the test sample to a focal group based on the nearest neighbor methods. Then, the learning model from the corresponding focal group will be used for the prediction of this test sample. The above example illustrates the multi-focal learning process and the reasons why it can lead to better performances for the data with the multi-focal property. However, in practice, a key developmental challenge is how to effectively identify the focal groups from the data.

3.3 Focal Group Formation: CORRELATION

As a practice, in this subsection, we propose a CORRELATION method to generate focal groups in problem logs. This method is motivated by our observation that problem logs have been provided by customers with diverse background. The experienced customers usually give more precise and focused descriptions about the problems. These problem descriptions are highly correlated with the problem solutions provided by service people. In contrast, the inexperienced customers usually give diverse descriptions for the same problem and their descriptions usually have low correlation with the final problem solutions.

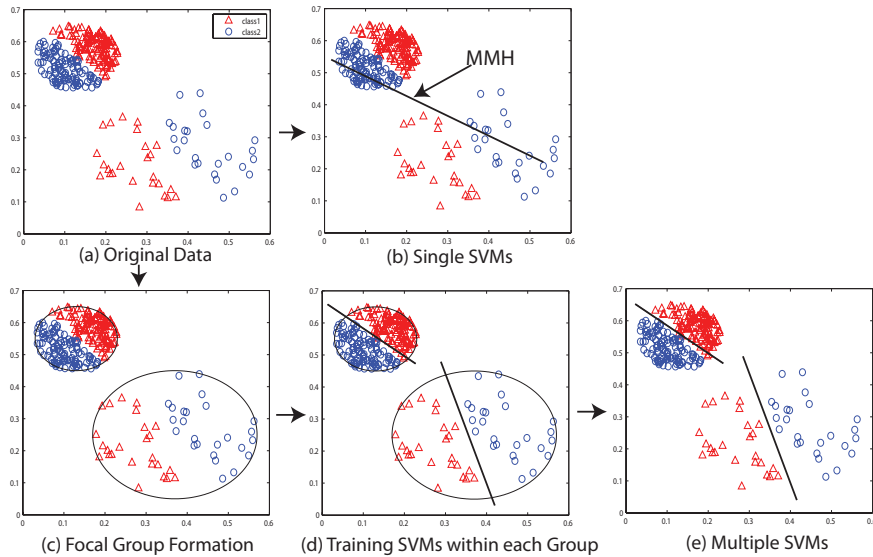


Figure 2: An Illustration of Multi-focal Learning.

To measure the correlation between problem descriptions and their problem solutions, we need to first transform problem descriptions and their solutions into vectors with 0/1 values. Specially, we first extract key words from the text descriptions of problems and their solutions and build a word vector to include all the key words. Then, a problem description or a problem solution can be transformed into a vector with 0/1 values based on whether the corresponding word in the word vector is in the problem description/solution or not. In this way, we turn all the problem descriptions and their problem solutions into binary vectors. To measure the strength of the relationships among these vectors, we use the Jaccard Coefficient [17], as follows.

$$\begin{aligned}
 J(X, Y) &= \frac{\#of\ matching\ presences}{\#of\ attributes\ not\ involved\ in\ 00\ matches} \\
 &= \frac{f_{11}}{f_{01} + f_{10} + f_{11}}.
 \end{aligned}
 \quad (1)$$

Where f_{ij} is the number of attributes where X is i and Y is j ($i = 0$ or 1 ; $j = 0$ or 1). After we compute the semantic correlation between problem descriptions and their problem solutions, we partition problem logs into two focal groups: one with high correlation values and another one with low correlation values. Note that we empirically choose the correlation thresholds for partitioning.

3.4 Focal Group Formation: ONTOLOGY

After carefully examining the problem logs, we have noticed that the key words from problem descriptions/solutions can be naturally organized into a hierarchy of concepts. In the above CORRELATION method, all the key words are treated equally in the correlation computation. However, the significance of key words from different concept levels should be different. Thus, in this subsection, we propose an ontology-enhanced correlation method (ONTOLOGY) for focal group formation in problem logs.

Indeed, ontology is an effective tool to represent hierarchical concepts within a domain [12] and it has been used for text classification [1]. In this study, we exploit ontology to improve focal group formation. Specifically, we construct an ontology with extracted concepts from problem logs. Figure 3 shows a sample ontology. In this ontology, the key

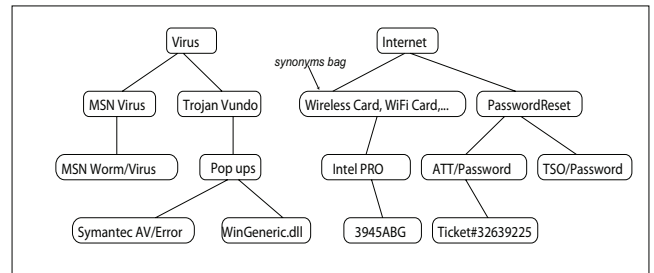


Figure 3: Sample Ontology.

words at top levels are very general and the key words at bottom levels are very specific. Both experienced and inexperienced customers are likely to use the key words from the top levels. However, the key words from the bottom levels (e.g. 3945ABG and Ticket#32639225) are usually the words hard-coded into the software system as error messages. Inexperienced customers are more likely to copy these key words when they report problems. In addition, there are some key words which are synonyms, such as monitor and LCD. Different customers may use different words for the same meaning. To capture these synonyms, we produce synonym bags which include all synonyms for the same concept. These synonym bags are integrated in the ontology.

With the help of the ontology, we know there is still room to better capture expert knowledge and improve focal group formation. Specifically, in ONTOLOGY, we first manually generate the ontology over all the key words. This process includes the generation of synonym bags. Then, we exploit a weighting scheme to assign small weights on the words at the top and the bottom levels and large weights on the words at the middle levels. Since the words at the middle levels are more likely used by experienced customers and the words at the top and bottom levels are more likely used by inexperienced customers, this weighting scheme can help to turn binary vectors into weighted vectors. In this case, we employ the cosine similarity to compute the correlations instead of the Jaccard measure, since the Jaccard measure can only handle binary vectors. Let X and Y be binary vectors of problem descriptions and solutions. Also, let W represent the weight vector obtained from ONTOLOGY.

OGY. Then the Ontology-Enhanced correlation can be computed as $Cos^w(X, Y) = \frac{(X*W) \cdot (Y*W)}{\|X\| \|Y\|}$, where $\|X\|$ is the length of vector X , \cdot is dot product and $*$ is defined as $X * W = [x_1 w_1, \dots, x_d w_d]$, where X, Y and W are both d dimension vectors.

Note that the ontology-enhanced correlation computing can increase the correlations between problem descriptions and their problem solutions for experienced customers and decrease the correlations between problem descriptions and their problem solutions for inexperienced customer, and thus better capturing two focal groups.

3.5 Risk Analysis of Multi-Focal Learning

In this subsection, we present a theoretical risk analysis of multi-focal learning. For the illustration purpose, we use the Naïve Bayes classifier as the base learning model.

First, given the input variable x and output variable y , the expected prediction error (EPE) [15] is:

$$EPE(f) := E[\mathcal{L}(y, f(x))] = \int_{x \times y} \mathcal{L}(y, f(x)) p(x, y) dx dy \quad (2)$$

where $\mathcal{L}(y, f(x))$ is the loss function, such as squared loss or 0-1 loss. The goal of classification is to find $f(x)$ so that EPE is minimized. In the Bayesian Classification terminology [7], EPE is also called $Risk$ which is actually more widely used than EPE . Assume that we have prior probability for class H_i as $p_i = p(H_i), i = 1, 2, \dots, M$. We can assign a loss function (\mathcal{L}_{ij}) to each possible decision outcome; We also know the conditional/likelihood probability as $p(x|H_i)$. Then, the total risk in the Bayesian terminology can be rewritten as:

$$\begin{aligned} Risk &= \sum_{j=1}^M \sum_{i=1}^M \mathcal{L}_{ij} p(H_i \text{ is true}) p(H_j \text{ is chosen} | H_i \text{ is true}) \\ &= \sum_{j=1}^M \sum_{i=1}^M \mathcal{L}_{ij} p_i \int_{R_j} p(x|H_i) dx \\ &= \sum_{j=1}^M \int_{R_j} \sum_{i=1}^M \mathcal{L}_{ij} p_i p(x|H_i) dx \end{aligned} \quad (3)$$

where R_j is the classification boundary. Since R_j partitions the entire space, any x belongs to exactly one such R_j . In fact, the Bayesian Classifier decides class j by maximizing the posterior probability; that is, $j := \operatorname{argmax}_j p(H_j|x)$

Then, the total risk can be simplified as

$$Risk = \sum_{i=1}^M \mathcal{L}_{ij} p_i p(x|H_i) \quad (4)$$

If we choose 0-1 criterion for loss function, Equation (4) can be rewritten as:

$$\begin{aligned} Risk &= \sum_{i=1, i \neq j}^M p_i p(x|H_i) = p(x) \sum_{i=1, i \neq j}^M \frac{p_i p(x|H_i)}{p(x)} \\ &= p(x) \sum_{i=1, i \neq j}^M p(H_i|x) = c \sum_{i=1, i \neq j}^M p(H_i|x) \end{aligned} \quad (5)$$

Since $p(x)$ is the same value for all x , the total risk is actually the product of one constant term c and the probability of error. Next, we introduce the following Theorem.

THEOREM 1. *The risk of multi-focal learning with the Naïve Bayes classifier is smaller than the risk of the single Naïve Bayes learning model.*

PROOF. Let $Risk'$ be the risk of multi-focal learning with the Naïve Bayes classifier and $Risk$ be the risk of the single Naïve Bayes learning. For a new object x , we assume that Class j' is predicted by the multi-focal learning model and Class j is predicted by the single learning model. According to Equation (5), we have $Risk' = \sum_{i=1, i \neq j'}^M p_i p'(x|H_i) = c \sum_{i=1, i \neq j'}^M p'(H_i|x)$ and $Risk = \sum_{i=1, i \neq j}^M p_i p(x|H_i) = c \sum_{i=1, i \neq j}^M p(H_i|x)$, where c is one constant number. We need to prove $\Delta Risk (\Delta Risk = Risk' - Risk) < 0$.

$$\begin{aligned} \Delta Risk &= Risk' - Risk \\ &= c \sum_{i=1, i \neq j'}^M p'(H_i|x) - c \sum_{i=1, i \neq j}^M p(H_i|x) \\ &= c(1 - p'(H_{j'}|x)) - c(1 - p(H_j|x)) \\ &= c(p(H_j|x) - p'(H_{j'}|x)) \end{aligned} \quad (6)$$

Apparently, both $p(H_j|x)$ and $p'(H_{j'}|x)$ are maximums as the posterior probability with the Naïve Bayes Classifier. Instead of $p'(H_{j'}|x)$, let us consider $p'(H_j|x)$ first. $p'(H_{j'}|x)$ will be bigger than $p(H_j|x)$ if $p'(H_j|x) > p(H_j|x)$, because $p'(H_{j'}|x)$ is maximal posterior probability for multi-focal learning model. That is, if $\Delta Risk^T = c(p(H_j|x) - p'(H_j|x)) < 0$, then $\Delta Risk = c(p(H_j|x) - p'(H_{j'}|x)) < 0$. So next we try to prove $\Delta Risk^T < 0$. For the Naïve Bayes Classifier, each attribute $x_k, k = 1, 2, \dots, D$, of x is assumed to be independent. In other words, $p(H_j|x) \propto p_j p(x|H_j) = p_j \prod_{k=1}^D p(x_k|H_j)$, where p_j is the same value for each class. So we can write $\Delta Risk^T$ as:

$$\begin{aligned} \Delta Risk^T &= c(p(H_j|x) - p'(H_j|x)) \\ &= c(p_j \prod_{k=1}^D p(x_k|H_j) - p_j \prod_{k=1}^D p'(x_k|H_j)) \\ &= c' (\prod_{k=1}^D p(x_k|H_j) - \prod_{k=1}^D p'(x_k|H_j)) \end{aligned} \quad (7)$$

where c' is also a constant number. Similar to the text classification problem, we use frequency ratio to estimate $p(x_k|H_j)$ or $p'(x_k|H_j)$. We have:

$$p(x_k|H_j) = \frac{(\# \text{ of } x_k | \text{ given } H_j)}{(\# \text{ of training samples} | \text{ given } H_j)} \quad (8)$$

$$p'(x_k|H_j) = \frac{(\# \text{ of } x_k | \text{ given } H_j)'}{(\# \text{ of training samples} | \text{ given } H_j)'} \quad (9)$$

where $\# \text{ of } x_k$ means the number of times of attribute x_k . Then, if $x_k \neq 0$, then x_k only appears in one focal group (this can be enforced by focal group formation methods), thus $(\# \text{ of } x_k | \text{ given } H_j)$ in Equation (8) and $(\# \text{ of } x_k | \text{ given } H_j)'$ in Equation (9) should be equal. However, $(\# \text{ of training samples} | \text{ given } H_j)'$ in Equation (9) is smaller than $(\# \text{ of training samples} | \text{ given } H_j)$ in Equation (8) for the whole training data. So $p'(x_k|H_j)$ is greater than $p(x_k|H_j)$ for $x_k \neq 0$. Also, if $x_k = 0$, the corresponding attribute tends to be irrelevant. For an irrelevant attribute, $p(x_k|H_j)$ or $p'(x_k|H_j)$ becomes almost uniformly distributed, thus it almost has no impact on the overall computation of posterior probability [17]; therefore, it has no impact on the value of $\Delta Risk^T$. As a result, we can conclude that $\prod_{k=1}^D p'(x_k|H_j)$ is greater than $\prod_{k=1}^D p(x_k|H_j)$, thus $\Delta Risk^T < 0$. So we can conclude $\Delta Risk < 0$. Thus, this theorem is held. \square

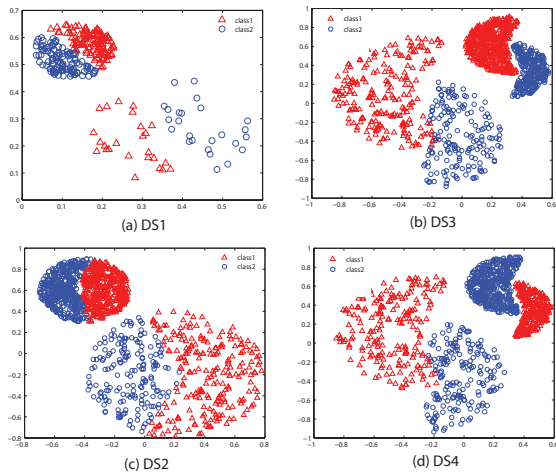


Figure 4: Synthetic Data Sets.

4. EXPERIMENTAL RESULTS

In this section, we provide an empirical study of the performances of multi-focal learning.

4.1 The Experimental Setup

Real-world Problem Logs. In the experiments, we have used real-world problem logs collected from IBM customer service centers. A detailed description of problem logs has been given in Section 2. In the experiments, we have formalized two-class classification as well as multi-class classification problems. For the two-class classification problem, we are focused on two categories of customer problems. One category includes the problems caused by the users, such as “reset/forget password”. Another category includes the problems related to product quality, such as “battery dead” and “system crash”. In contrast, to study the multi-class learning problem, we prepare three categories of problems including user-caused problems, hardware problems, and software problems. All the problems were labeled by domain experts. Finally, to evaluate the focal group formation methods, problem descriptions were also labeled as “experienced” or “inexperienced” by domain experts. These labels are used as benchmarks in the experiments.

Synthetic Data. Figure 4 shows four synthetic data sets including DS1, DS2, DS3 and DS4, which have the multi-focal property. Some data characteristics are shown in Table 2.

Table 2: Some Characteristics of Synthetic Data

dataset	# of classes	size of class1	size of class2
DS1	2	131	128
DS2	2	617	554
DS3	2	662	409
DS4	2	614	459

Experimental Tools. We employ four base classifiers, Support Vector Machines (SVMs), Naive Bayes classifier, decision tree, and rule-based classifiers (RIPPER [18]). Also, we use MFL to indicate multi-focal learning.

To evaluate focal group formation methods, we compare CORRELATION and ONTOLOGY with clustering and random partitioning methods. For the clustering method, we employ Chameleon [5] in CLUTO [10], since we need to capture clusters with different densities. Problem descriptions labeled by domain experts are used as the benchmark.

Evaluation Metrics. The classification accuracy and F-measure [17] have been used for the performance evaluation. For all the experiments, we did five-cross validation.

4.2 Results on Problem Logs

In this subsection, we show the performances of multi-focal learning on real-world problem logs.

Two-class Categorization. The goal is to group problems into two categories: *user-caused* problems and *product* problems. Here, we use CORRELATION for generating focal groups. Specifically, we measure the correlations between problem descriptions and their problem solutions. If the correlation is lower than 0.07, the corresponding problem log is placed in a sparse (inexperienced) group. If the correlation is larger than 0.1, we put the problem log in a dense (experienced) group. The thresholds are empirically specified. Once we have two focal groups, we build learning models with some base classifiers on both focal groups.

Table 3: Performances of MFL (2 Classes)

Method	Accuracy within group	Accuracy
SVMs	N/A	59.43%
MFL with SVMs	68.24% (dense group)	71.94%
	74.59% (sparse group)	
Ripper	N/A	67.14%
MFL with Ripper	74.71% (dense group)	77.74%
	79.89% (sparse group)	
C4.5	N/A	64.50%
MFL with C4.5	71.28% (dense group)	77.82%
	82.70% (sparse group)	
Bayes	N/A	67.60%
MFL with Bayes	61.24% (dense group)	68.84%
	73.59% (sparse group)	

Table 3 shows the comparison results of multi-focal learning and traditional classification methods including SVMs, Ripper, C4.5, and Bayes. In the table, we can observe that multi-focal learning can improve the performances of these traditional classification methods with a significant margin.

Table 4: Performances of MFL (3 Classes)

Method	Accuracy within group	Accuracy
SVMs	N/A	44.15%
CORRELATION		
MFL with SVMs	72.38% (dense group)	52.86%
	39.25% (sparse group)	
ONTOLOGY		
MFL with SVMs	72.62% (dense group)	53.17%
	39.88% (sparse group)	
Bayes	N/A	48.39%
CORRELATION		
MFL with Bayes	55.90% (dense group)	51.05%
	47.72% (sparse group)	
ONTOLOGY		
MFL with Bayes	55.01% (dense group)	51.29%
	48.83% (sparse group)	

Three-class Categorization. Here, we target grouping problem logs into three categories: *user-caused problems*, *hardware problems*, and *software problems*. In this experiment, we apply both CORRELATION and ONTOLOGY methods for generating focal groups. Table 4 shows the comparison results of multi-focal learning and traditional classification methods including SVMs and Bayes. As can be seen, multi-focal learning can improve the classification performances of both SVMs and Bayes. Also, we can observe that ONTOLOGY leads to slightly better classification performances compared to CORRELATION.

4.3 Performance Comparisons

In this subsection, we evaluate the performances of several focal group formation methods including CORRELATION, ONTOLOGY, clustering methods, random partitioning on

real-world problem logs. Here, we use SVMs as the base classifier and target two-class categorization problem, which has a similar experimental setting as the two-class categorization problem in Section 4.2.

For ONTOLOGY, we stratify the domain concepts into four levels. The weights on concepts on the top and bottom levels are set as one and the weights on concepts on the middle two levels are set as three. Then, we measure weighted correlations between problem descriptions and their problem solutions. If the correlation is lower than 0.13, the corresponding problem log is placed in a sparse group. If the correlation is greater than 0.23, the problem log is placed in a dense group. These two thresholds are empirically specified. In addition, the clustering method we used is the Chameleon algorithm in CLUTO, since there are different densities in the data. The default parameters for running Chameleon in CLUTO were used except the number of neighbors (-nbnrs=80). Other parameters for Chameleon include the use of graph clustering method (-clmethod=graph) with correlation (-sim=corr) as the similarity and the use of agglomeration (agglofrom=30). Finally, we use the labels from domain experts (EXPERT) as the benchmark.

Table 5: Performance Comparisons

Method	Accuracy within group	Accuracy	Class	Fmeasure
SVMs	N/A	59.43%	1	0.7344
			2	0.1119
CORRELATION				
MFL	68.24% (dense group)		1	0.6855
	74.59% (sparse group)	71.94%	2	0.7076
ONTOLOGY				
MFL	69.41% (dense group)		1	0.6718
	75.54% (sparse group)	73.01%	2	0.7139
CLUSTERING				
MFL	67.10% (dense group)		1	0.7338
	56.17% (sparse group)	59.45%	2	0.1277
RANDOM				
MFL	47.09% (group 1)		1	0.7326
	65.96% (group 2)	58.58%	2	0.0982
EXPERT				
MFL	71.27% (dense group)		1	0.7767
	74.96% (sparse group)	73.53%	2	0.6644

Table 5 shows the results. As can be seen, the performance of ONTOLOGY is slightly better than that of CORRELATION and is closer to the performance by domain experts as indicated by EXPERT in the table. Also, both ONTOLOGY and CORRELATION can lead to a much better classification performance compared to the clustering method and random partition. The above indicates that both CORRELATION and ONTOLOGY are effective methods for identifying focal groups in problem logs.

4.4 Results on Synthetic Data

In this subsection, we exploit some synthetic data to better illustrate the multi-focal property; that is, the impact of the diversity inherent in the data on the learning performance. For four synthetic data sets shown in Figure 4, we can see that there are simple linear-separable concepts as well as complex non-linear-separable concepts [9] in these data sets. Indeed, multi-focal learning is a natural solution to the data with complex non-linear-separable concepts, because multi-focal learning allows for decomposing the complex concepts into simple linearly separable concepts by grouping data objects into different focal groups.

Since two-dimensional synthetic data sets in Figure 4 can be easily decomposed by the clustering algorithms, we apply

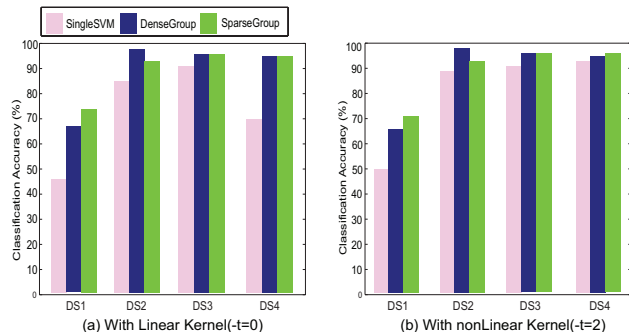


Figure 5: Accuracy Performances on Synthetic Data.

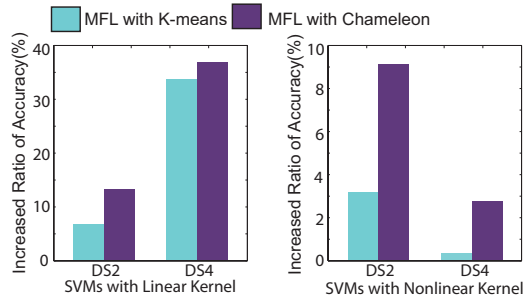


Figure 6: MFL with Clustering Methods.

the clustering methods for generating focal groups. Specifically, we use Chameleon [5] in CLUTO [10] with parameters: -clmethod=graph (graph clustering), -sim=corr (correlation similarity), and -nbnrs=60 (60 nearest neighbors).

Table 6 shows the results. Here, we use SVMs with both linear and non-linear kernels as the base classifiers. As can be seen, in terms of F-measure, the performances of multi-focal learning are much better than that of single SVMs no matter linear or non-linear kernels are used in SVMs. Note that we only show the results of SVMs with linear kernels on DS1 and DS2 data sets and the results of SVMs with non-linear kernels on DS3 and DS4 data sets. A similar trend has actually been observed on all four synthetic data sets. Due to the space limitation, we omit these results. In addition, Figure 5 shows the learning performances in terms of the classification accuracy. In the figure, we can see that the performances of both dense group and sparse group are improved significantly with multi-focal learning.

Another interesting observation in Figure 5 is that, while the clustering algorithms generate the same focal groups for DS3 and DS4, the distributions of their class objects are different. Indeed, for SVMs with linear kernel, the single model has a much worse performance on DS4 than on DS3. This indicates that the complex concepts in DS4 have a negative impact on the performance of linear kernels. However, with multi-focal learning, the performance of SVMs with linear kernels on DS4 has been improved a lot.

Finally, we show the reasons why we choose Chameleon for data with different densities instead of widely-used K-means. Here, we use both K-means and Chameleon for generating focal groups on DS2 and DS4 data sets. Figure 6 shows the increased ratio of classification accuracies by multi-focal learning. As can be seen, for both linear and non-linear cases, Chameleon leads to much better performances than K-means, while K-means can also lead to an improved performance of multi-focal learning.

Table 6: F-Measure Comparisons on Synthetic Data

SVMs with linear kernel(-t=0)					
Data	Method	class	Precision	Recall	F-measure
DS1	SVMs	1	0.5106	0.8276	0.6316
		2	0.4444	0.1481	0.2222
	MFL	1	0.8744	0.7800	0.7520
		2	0.8463	0.8092	0.7580
DS2	SVMs	1	0.8443	0.8638	0.8535
		2	0.8441	0.8210	0.8318
	MFL	1	0.9577	0.9575	0.9571
		2	0.9543	0.9507	0.9518
SVMs with nonlinear kernel(-t=2)					
DS3	SVMs	1	0.9168	0.9321	0.9243
		2	0.8867	0.8628	0.8743
	MFL	1	0.9678	0.9548	0.9612
		2	0.9279	0.9486	0.9380
DS4	SVMs	1	0.9868	0.8400	0.9072
		2	0.8919	0.9919	0.9391
	MFL	1	0.9510	0.9248	0.9375
		2	0.9442	0.9641	0.9539

4.5 Classification Error Analysis: A Case Study

Here, we provide a simple case study to show that multi-focal learning with a Naive Bayes classifier can lead to smaller classification errors. First, let us consider a binary classification problem, where y_1 and y_2 represent two classes and X represents objects. By Bayesian Theory, we have:

$$p(y/X) = \frac{p(X/y)p(y)}{p(X)} \quad (10)$$

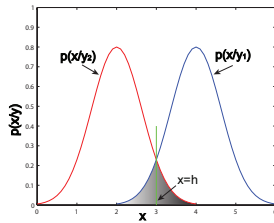


Figure 7: One Dimension Class-Conditional Probabilities.

To make decision for classification, it is equal to compare the posterior probabilities of $p(y_1/X)$ and $p(y_2/X)$. In Equation (10), $p(X)$ is equal for all objects. Furthermore, if we assume $p(y_1)$ is equal to $p(y_2)$, then $p(y/X)$ in Equation (10) can be rewritten as $p(y/X) = cp(X/y)$, where $c = p(y)/p(X)$. Thus, we get $p(y_1/X) = cp(X/y_1)$ and $p(y_2/X) = cp(X/y_2)$. Next, to classify objects, we only need to compare class conditional probabilities $p(X/y_1)$ and $p(X/y_2)$. There are several ways to estimate these conditional probabilities by exploiting training data [17]. Here, we take the Gaussian Function for estimating conditional probability. To demonstrate the classification error, we present the class conditional probabilities and classification boundary in one dimension as shown in Figure 7, where $x = h(X = x \text{ in one dimension})$ is the classification boundary for these two classes. The classification error is

$$error = \int_{-\infty}^h p(x/y_1)dx + \int_h^{\infty} p(x/y_2)dx. \quad (11)$$

According to Figure 7 or Equation (11), we know that the classification error is proportional to the overlap area of these two Gaussian functions. As we know, the overlap area is determined by the means and variances of these two Gaussian functions. Similar to the use in Section 3.2, we use

two-dimension Gaussian functions to estimate conditional probabilities of the synthetic data introduced in Section 3.2. Specifically, for single model we use two Gaussian functions, Gau_1 and Gau_2 , to estimate the conditional probabilities of two classes. For multi-focal learning, we need four Gaussian functions, Gau_1^D and Gau_2^D for the dense group and Gau_1^S and Gau_2^S for the sparse group. We show these Gaussian functions in Figure 8. As can be seen, there is more overlap between Gau_1 and Gau_2 than that between Gau_1^D and Gau_2^D or Gau_1^S and Gau_2^S . Since this kind of overlap is an overlap across three dimensions, to make it more clear, we also show the projection of the overlap on the two-dimension space as shown in Figure 8. In this case, the classification error can be computed by integrating over the two dimension area as the following:

$$error = \int_{R_1} \int_{R_1} Gau_1 dX + \int_{R_2} \int_{R_2} Gau_2 dX \quad (12)$$

where, R_1 and R_2 are determined as

$$\begin{aligned} R_1 &:= \{X | Gau_1(X) < Gau_2(X)\} \\ R_2 &:= \{X | Gau_2(X) < Gau_1(X)\} \end{aligned} \quad (13)$$

Note that, for the dense group and the sparse group, both Equation (12) and Equation (13) have the same form. To this point, we can clearly see that classification error of multi-focal learning is much smaller than the error produced by the single model by either the Gaussian-function figure or the error computation in Equation (12).

5. CONCLUSIONS AND DISCUSSIONS

In this paper, we formalized a multi-focal learning problem, which was motivated by the observations of diversities of samples in training data. The key idea of the multi-focal learning is to divide training data into different focal groups and the learning models should be learned within each focal group instead of building a single learning model using all the training data as a whole. The multi-focal learning allows the learning algorithms to mitigate the influence of the diversities inherent in training data, and thus leads to better learning performances.

As a practice, we have exploited the multi-focal learning techniques for automatic problem categorization in real-world problem logs collected from customer service centers. A critical challenge in the multi-focal learning is how to identify focal groups in training data. To address this challenge in problem logs, we proposed a correlation method (CORRELATION) to partition problem descriptions within each class into two different groups: one for experienced customers and the other for inexperienced customer. In addition, to better capture the information encoded in problem logs, we also developed an ontology-enhanced correlation method (ONTOLOGY) for identifying different focal groups. Experimental results show that both CORRELATION and ONTOLOGY have led to better learning performances than other focal-group formation methods, such as the methods based on clustering and random-partition, while the learning performance by ONTOLOGY is lightly better than that by CORRELATION.

Discussions. In this study, we have illustrated the concept of multi-focal learning by exploiting problem logs collected in real-world customer service centers. While the solution for forming multiple focal groups has made use of data

