Automatic Classification of Change Requests for Improved IT Service Quality

Cristina Kadar, Dorothea Wiesmann, Jose Iria, Dirk Husemann, Mario Lucic
IBM Zurich Research Laboratory
Säumerstrasse 4, CH-8803 Rüschlikon, Switzerland
{*cka,dor,jir,hud,cic*}*@zurich.ibm.com*

***Abstract*—Faulty changes to the IT infrastructure can lead to critical system and application outages, and therefore cause serious economical losses. In this paper, we describe a change planning support tool that aims at assisting the change requesters in leveraging aggregated information associated with the change, like past failure reasons or best implementation practices. The thus gained knowledge can be used in the subsequent planning and implementation steps of the change. Optimal matching of change requests with the aggregated information is achieved through the classification of the change request into about 200 fine-grained activities. We propose to automatically classify the incoming change requests using various information retrieval and machine learning techniques. The cost of building the classifiers is reduced by employing active learning techniques or by leveraging labeled features. Historical tickets from two customers were used to empirically assess and compare the accuracy of the different classification approaches (Lucene index, multinomial logistic regression, and generalized expectation criteria).**

***Keywords*-service quality; change management; automation; text classification; information retrieval; logistic regression; generalized expectation criteria;**

I. Introduction

In the past years IT Service Management (ITSM) has become an important field of research to support IT Service providers in their quest for delivering higher service quality with ever increasing efficiency. A promising approach applied in a number of studies is to transpose methodologies and frameworks that have yielded tremendous efficiency and quality gains in the manufacturing industry. While the adoption of a global delivery model by IT service providers enabling standardization, economies of scale, and cost optimization has subsequently led to increased delivery efficiencies, the application of quality improvement frameworks successfully deployed in manufacturing is only in its beginnings [1]. In addition, to account for the large human component in service delivery, efficient knowledge sharing among service provider personnel is a promising complementary means to increasing IT service delivery.

Previously the IT Infrastructure Library (ITIL) V2, the de-facto standard for concepts and practices for Information Technology Services Management, merely advocated knowledge sharing in the area of incident and problem management, e.g. through the establishment of a "Known Error Database". However, its latest version (V3) reflects the service quality improvement trends by adding both the Continual Service Improvement (CSI) practice and a Knowledge Management Process [2], [3].

Motivated by the observation that faulty changes have been found to cause over 60% of critical system and application outages [4], our research has focused on applying continuous improvement and knowledge management concepts to the reduction of failed IT changes. In particular, we have devised and developed a tool, that assists a change requester in understanding past failure reasons for similar changes and in learning about best implementation practices. Towards this end, our tool automatically classifies a new change ticket and retrieves aggregated data from various resources associated with the change category. In this paper, we describe the applied change ticket classification scheme. The remainder of the paper is structured as follows. A brief review of related work on ticket classification is given in the next section. Section III presents an overview of the system. In Section IV we introduce the categorization methods we applied to our classification problem. A complete description of the experimental setting is given in Section V, and in Section VI a comparative evaluation of the methods is presented, followed by a discussion on the results obtained. We conclude with a mention to our plans for future work in Section VII.

II. Related Work

Our paper bears similarities with a number of publications on maintenance and incident ticket classification [5]–[7]. Lucca and coworkers have applied automatic classification of user generated error-reporting logs, i.e. software maintenance tickets, with the goal to automate assignment of the tickets to the appropriate team of the maintenances provider. Similar to our work, the texts to be classified are short, natural language statements; however the number of classes is significantly lower than ours, namely eight compared to up to 200, respectively. Comparing various classification methods, Lucca et al. find that probabilistic models perform best and, unlike other methods, increase precision with a growing training set. They further observe that expert feedback increases the accuracy of the probabilistic model by 7% [5].

Gupta's and coworkers' publication is based on a similar scenario as ours: they describe the automatic classification of an incident ticket as part of the record creation in the

978-0-7695-4371-0/11 $26.00 © 2011 IEEE
DOI 10.1109/SRII.2011.95

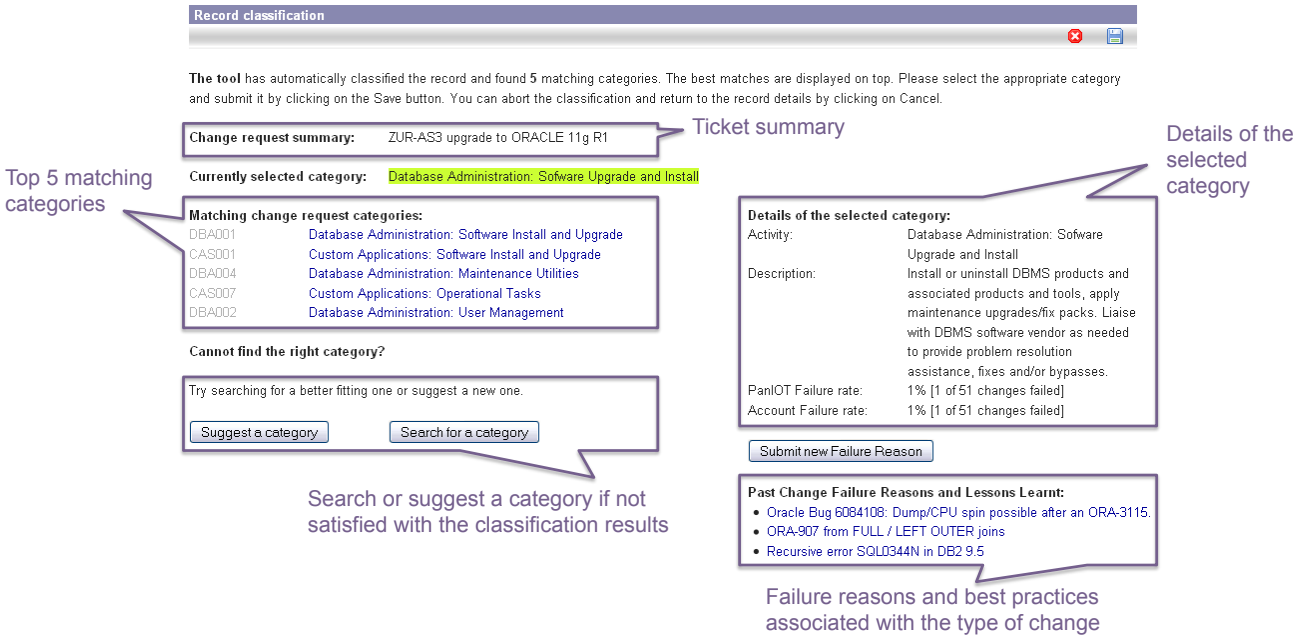


Figure 1. Classification screen.

IBM Tivoli Service Request Manager[1] by a person with limited expertise. As in our case, the alternative deployed so far is a manual selection of one of many pre-configured classes which is time consuming and error-prone [6]. The incident tickets are either user-generated or system-generated and thus Gupta et al. deploy two different classification approaches, namely a machine learning approach (Naïve Bayesian) and a rule-based approach, respectively. In addition they allowed experts to bootstrap and/or complement the machine generated features. The resulting incident classifier achieved 70% accuracy with 1000 features.

In contrast to the above mentioned publications, in the application scenario described by Diao and coworkers, sets of incident tickets are classified into failure codes after the incidents are resolved to facilitate root cause analysis and failure trend monitoring [7]. Anticipating a number of issues when applying machine-learning techniques for classification, namely cost of labeling, poor quality of incident tickets, and inconsistency in tickets caused by the global deployment, the authors have implemented a rule-based classification and compared its accuracy with that of a Naïve Bayesian classifier. In their specific setting, in which all three expert system roles are perform by the same community of quality analysts, the rule-based classifier outperforms the Naïve Bayesian classifier both for smaller and larger data sets. Additionally, initial tests reveal a lower cost of creating the rule set compared to the cost of labeling the tickets.

[1]http://www-01.ibm.com/software/tivoli/products/service-request-mgr

III. System

Our change preparation support tool implements three process steps: (1) the change requester enters the change ticket information; (2) the tool classifies the change request into the catalogue categories and displays the top five matching categories; if no matching category was found in the previous step, or if the user is not satisfied with the returned list, he can browse all categories for the appropriate one; (3) the change requester selects the best matching category and reviews past failure reasons/known issues as well as best practices for the selected class of changes. The thus gained knowledge can be used in the subsequent planning and implementation steps to minimize the failure risk. The classification screen, illustrating the process for a database management related change, is shown in Figure 1.

A. Categories

The central artifacts of our approach are the categories along which the changes are organized. The classification schema is based on a catalogue of standard activities performed to service outsourced IT infrastructure. We use up to 200 fine-grained activities. This high granularity is necessary to ensure the relevance of the retrieved information to the change requester but at the same time is making the automatic classification a non-trivial task. On the other hand, a category is more than just a label without meaning, as it comes with various descriptive fields. Two of these fields are the name and the description of the catalogue activity, both in natural language. Moreover, activities from the same technical area of services are grouped together into a service

line. We cover lines like database management, network services, Unix servers management, etc.

B. Tickets

The change ticket is a record of the planned change in an ticketing system that supports the change management work flow. It contains information necessary to approve, classify, and implement the change. ITIL prescribes the following fields: unique ID, change initiator, date of submission, short summary of the change to be implemented, reason for the change to be implemented, business and IT infrastructure impact, risks, priority, schedule, and resources. Some of the fields are in natural language, e.g. the short summary and reason for change, and some are predefined multiple-choice fields. Some of the fields are filled in automatically and not all of them are compulsory. Moreover, change tickets are assigned to different work groups with different specialization [2]. In the remainder of the paper we call these technical expert groups responsible for the change owner groups.

IV. Automatic Classification

We face a unique and challenging classification set-up. Firstly, the large amount of very granular, often similar categories poses a real threat to the success of any categorization attempt. In contrast, the extra information we have on categories can be exploited to bootstrap learning, e.g. the fact that categories belong to certain service lines that can be mapped to the owner group feature of a ticket instance. Secondly, pre-classified tickets for training are a rare commodity but historical unlabeled tickets are easily obtainable, as the customers had already been tracking their changes. The laborious task of annotating old tickets requires a skilled domain expert and represents the most costly aspect of building a classifier. Thirdly, each customer has its own lingua for describing changes and its specific distribution over the classes, but the dissemination of learning across customers is desired. Finally, user's choice and feedback can be saved and used in assessing and rebuilding the classifier.

The following subsections present the approaches we have explored to tackle these classification challenges.

A. Information Retrieval Approach

Seeing the classification problem as an information need (*classify* $\stackrel{\text{def}}{=}$ *retrieve the matching category*), we use a text search engine and build an index of all categories in the schema. For each of the categories, we save its relevant fields and additionally, keywords defined by experts of the customer. Incoming change requests are seen as queries that get scored against this static index. Returned is a ranked list of matching categories.

We use Lucene[2], an open-source text search engine library, to create and read the index.

[2]http://lucene.apache.org

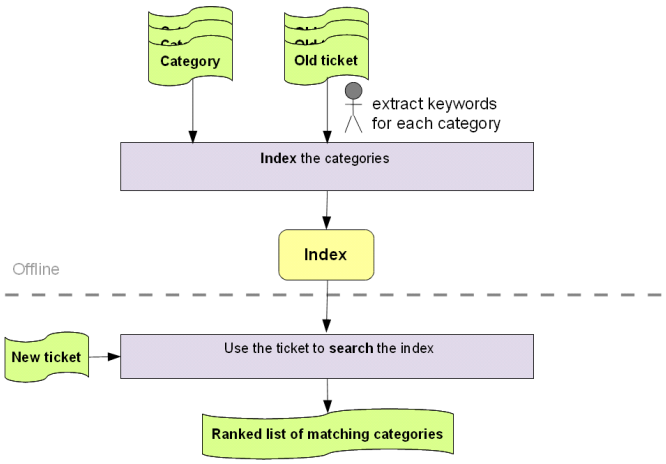


Figure 2. Information retrieval setup.

Lucene extends the classical Vector Space Model (VSM) of Information Retrieval [8]. In VSM, documents (in our case: categories) and queries (in our case: tickets) are represented as weighted vectors in a multi-dimensional space, where each distinct index term is a dimension, and weights are *tf-idf* [9] values. According to this metric, the weight of the j-th term of the vocabulary in the i-th document is derived from the *term frequency* $tf_{i,j}$ of the term in the document, and the *inverse document frequency* idf_j of the term over the entire set of documents, as follows:

$$tf\text{-}idf_{i,j} = tf_{i,j} \cdot \log(idf_j)$$

The VSM score of document d for query q is the cosine similarity of the weighted query vectors $V(q)$ and $V(d)$:

$$sim(q,d) = \frac{V(q) \cdot V(d)}{|V(q)| \cdot |V(d)|}$$

where $V(q) \cdot V(d)$ is the dot product of the weighted vectors, and $|V(q)|$ and $|V(d)|$ are their Euclidean norms.

Lucene refines the VSM score for both search quality and usability. At index time, users can specify that certain documents are more important than others or that certain fields weight more, by assigning a document field boost $doc\_boost(d)$. At search time, users can specify boosts to each query or each query term, hence the contribution of a query term to the score of a document is multiplied by the boost of that query term $query\_boost(q)$. Aside from that, a document may match a multi-term query without containing all the terms of that query, and users can further reward documents matching more query terms through a coordination factor, which is usually larger when more terms are matched: $coord\_factor(q,d)$.

Based on the query q, the score of document d is com-

puted by the following (simplified) formula:

$$score(q,d) = sim(q,d) \cdot doc\_boost(d) \cdot query\_boost(q) \cdot coord\_factor(q,d)$$

Using this method, satisfactory accuracy levels in top 5 returned categories are recorded (refer to Section VI for the exact results). However, because of the varying vocabulary, a new set of keywords needs to be defined every time a new customer is on-boarded. This makes the method not scale-up.

B. Machine Learning Approach

As an alternative to the above approach, we use machine learning techniques [10], which are able to automatically discover discriminative features and learn over time and across related domains. In the next subsection we explore two different classification settings.

1) Supervised Setting: In a supervised setting, an inductive process automatically builds a model by learning from a set of pre-classified documents (in our case: tickets). The classifier is built offline and its quality is estimated by looking at the error rate when tested on another set of pre-classified documents. To assure the correct validation, it is vital that these documents were not seen in the learning phase. As shown in Figure 3, the model will then be employed online to score every new incoming ticket. A ranked list of matching categories is the final output. Rather than allocate every unique word in the training corpus to a distinct feature, we can optionally perform feature selection [11]. By reducing the dimensionality of the input, we enhance the speed and the generalization capability of the learning algorithm.

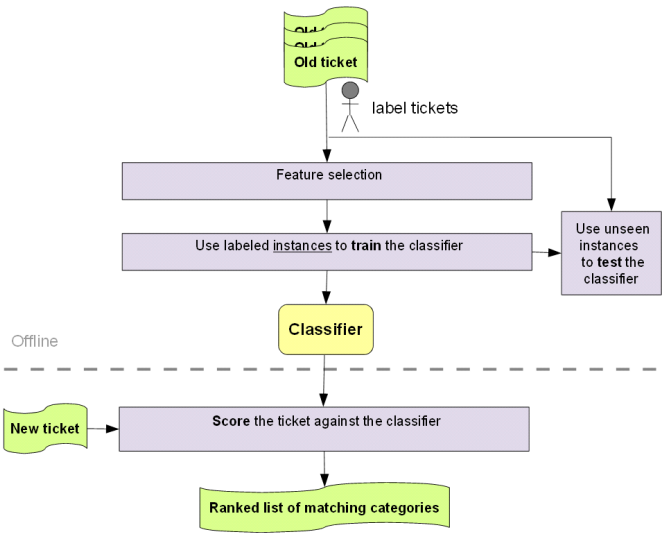


Figure 3. Supervised learning setup.

We choose to implement a regularized logistic regression as it has been shown to provide outstanding predictive performance across a range of text classification tasks and corpora [12], [13]. The *multinomial logistic regression* classifier (also known in the literature as the maximum entropy classifier) is the generalization of this model to multi-class classification.

Below, we deduct the classification function for the multi-class, single-label classification case. Let $\mathbf{x}$ be the input (a vector of features $\mathbf{x}_j$), y the output (a discrete class label), and θ the parameters of the model. The probability of output y conditioned on input $\mathbf{x}$ is given by:

$$p(y|\mathbf{x};\theta) = \frac{exp(\sum_j \theta_j \mathbf{x}_j)}{Z(\mathbf{x})}$$

where $Z(\mathbf{x}) = \sum_y exp(\sum_j \theta_j \mathbf{x}_j))$ is a normalizer which ensures that $\sum_y p(y|\mathbf{x};\theta) = 1$.

To learn the particular values of θ (in other words, the weights of all features for each category), we compute the log-likelihood of the labels on the training data $\mathcal{T}$:

$$L(\theta|\mathcal{T}) = \sum_{(\mathbf{x},y)\in\mathcal{T}} \log(p(y|\mathbf{x};\theta))$$

The method of maximum likelihood estimates $\hat{\theta}$ by finding a value of θ that maximizes the log-likelihood:

$$\hat{\theta} = \arg\max_{\boldsymbol{\theta}} L(\theta|\mathcal{T})$$

So far the model implicitly assumes a uniform prior distribution of parameters and can suffer from over-fitting. By imposing as regularizer a univariate Gaussian prior with mean $\mu = 0$ and variance σ^2 on the parameters, the model will now prefer more uniform models. This way, overfitting can be reduced and performance improved.

The final objective function to be maximized is:

$$\mathcal{O} = \sum_{(\mathbf{x},y)\in\mathcal{T}} \log(p(y|\mathbf{x};\theta)) - \sum_j \frac{\theta_j^2}{2\sigma^2}$$

To speed-up the learning task, we employ *active learning*. The key idea behind active learning is that a machine learning algorithm can achieve greater accuracy with fewer training labels if it is allowed to choose the data from which it learns [14]. It is useful in settings where unlabeled data may be abundant or easily obtained, but labels are difficult, time-consuming, or expensive to obtain.

Pool-based active learning [15] assumes there is a small set of labeled data $\mathcal{L}$ and a large pool of unlabeled data $\mathcal{U}$ available. The queries are selected from the pool of unlabeled instances using an uncertainty sampling query strategy, which selects the instance in the pool about which the model is least certain how to classify.

There have been many proposed ways of formulating such query strategies in the literature. We choose to use a multi-class uncertainty sampling variant called *margin sampling* [16].

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} p(\mathbf{x}|\hat{y_1};\theta) - p(\mathbf{x}|\hat{y_2};\theta)$$

where $\hat{y_1}$ and $\hat{y_2}$ are the first and the second most probable class labels under the model, respectively. Intuitively, instances with large margins are easy, since the classifier has little doubt in differentiating between the two most likely class labels. Instances with small margins are more ambiguous, thus knowing the true label would help the model discriminate more effectively between them.

In this supervised setting and with a large amount of training data, the classifier can achieve very high accuracies for one customer (refer to Section VI for the exact results). However, applying the same model to a different customer results in a dramatic drop in accuracy. This means that a training set of pre-labeled tickets is needed for each new customer, which is very expensive in terms of time and money. Whenever a new customer is on-boarded, laborious work is required of the domain experts, the ones responsible for training the model before deployment for use by the change requesters.

2) Semi-Supervised Setting with Weakly-Labeled Data: A setup which uses weakly labeled data ("side-information") would allow us to automatically exploit the intention of each class (that means, the extra information we have on the class, like description or service line) and to point out any other affinities between the input features and the classes.

Instead of learning from labeled instances (i.e. documents), the classifier learns this time from labeled features (i.e. words). Figure 4 reflects the steps taken in building such a classifier. First, the algorithm identifies a relatively small set of words in the unlabeled corpus that are both highly predictive of some class(es), and occur often enough to have impact. Second, some of these words are automatically labeled, whereas the others are presented to the user. For each word, the user can choose to label it, that is, deliver a class or a list of classes for which the respective feature is a strong indicator. Alternatively, the user can choose to reject the word, if he feels it is not a strong feature. Finally, based solely on this information, a classifier is built. Similar to the previous setup, new tickets are rated online against this model.

Rather than requiring documents in the training to be examined and labeled, our approach leverages a small set of words that domain experts indicate to be positively correlated with each class – the labeled features. We adopt the *generalized expectation criteria* method [17], [18] to translate this kind of domain knowledge into constraints on model expectations for certain word-class combinations.

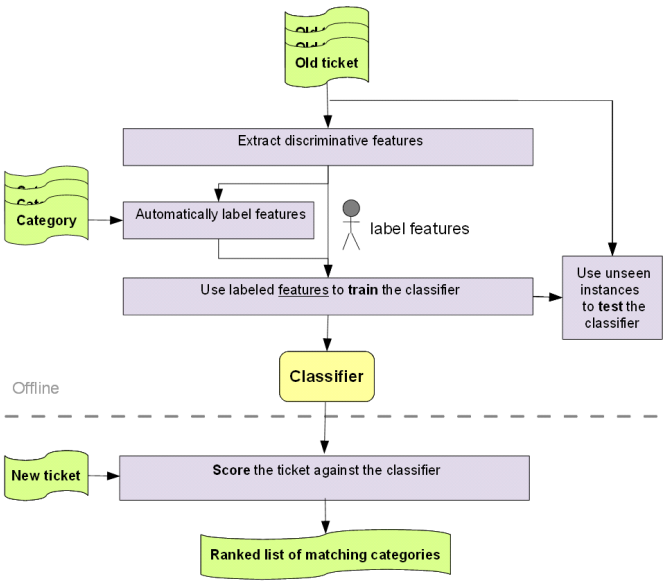


Figure 4. Semi-supervised learning setup with weakly-labeled data.

A generalized expectation (GE) criterion is a term in a parameter estimation objective function that assigns scores to values of a model expectation. Again, let $\mathbf{x}$ be the input, y the output, and θ the parameters for a given model. Given a set of unlabeled data $\mathcal{U} = \{\mathbf{x}\}$ and a conditional model $p(y|\mathbf{x};\theta)$, a GE criterion $G(\theta;\mathcal{U})$ is defined by a score function V and a constraint function $G(\mathbf{x}, y)$:

$$G(\theta;\mathcal{U}) = V(E_{\mathcal{U}}[E_{p(y|\mathbf{x};\theta)}[G(\mathbf{x}, y)]]).$$

The GE formulation is generic enough to enable exploring many different choices of score functions and constraint functions. In this paper, we maximize the GE term together with an entropy regularization term in the objective function, although this can be easily combined with an empirical loss term to form a composite objective function that takes into account labeled instances as well. Moreover, we use label regularization, that is, the constraints are expectations of model marginal distributions on the expected output labels. As such, we use estimated label marginal distributions $\tilde{g}_{\mathbf{x},y} = \tilde{p}(y)$ and consider constraints of the form $G(\mathbf{x}, y) = \vec{1}(y)$. Model divergence from these constraints can be computed by using, for example, KL-divergence [19]:

$$G(\theta;\mathcal{U}) = -D(\tilde{p}(y)||E_{\mathcal{U}}[\vec{1}(y)p(y|\mathbf{x};\theta)]).$$

We compute the model divergence by:

$$G(\theta;\mathcal{U}) = -\sum_{i \in F(\mathcal{U})} D\left(\hat{p}(y|\mathbf{x}_i > 0)||\tilde{p}_\theta(y|\mathbf{x}_i > 0)\right)$$

where F is a function that returns the set of features in the input data, $p(y|\mathbf{x}_i > 0) = \frac{1}{C_i}\vec{1}(y)\vec{1}(\mathbf{x}_i > 0)$ is an

indicator of the presence of feature i in x times an indicator vector with 1 at the index corresponding to label y and zero elsewhere, and $C_i = \sum_{\mathbf{x}} \vec{1}(\mathbf{x}_i > 0)$ is a normalizing constant; $\tilde{p}_\theta$ denotes the predicted label distribution on the set of instances that contain feature i and $\hat{p}$ are reference distributions derived from the labeled features.

We estimate the reference distributions using the method proposed by [20]: let there be n classes associated with a given feature out of L total classes; then each associated class will have probability q_{maj}/n and each non-associated class has probability $(1 - q_{maj})/(L - n)$, where q_{maj} is set by the domain experts to indicate the correlation between the feature and the class.

To encourage the model to have non-zero values on parameters for unlabeled features that co-occur often with a labeled feature, we select again as regularizer the Gaussian prior on parameters, which prefers parameter settings with many small values over settings with a few large values. The combined objective function is finally:

$$\mathcal{O} = - \sum_{i \in F(\mathcal{U})} D\left(\hat{p}(y|\mathbf{x}_i > 0) || \tilde{p}_\theta(y|\mathbf{x}_i > 0)\right) - \sum_j \frac{\theta_j^2}{2\sigma^2}$$

V. Experimental setup

We use two different datasets of historical tickets to experimentally evaluate the proposed approaches. The collection from the first customer, called in the following customer A, contains 1317 tickets. Tickets in this dataset span across 54 different categories (see Figure 5a), collected from December 2009 to July 2010. The summaries of these tickets are compact, very technical pieces of text – the whole vocabulary consists of only 985 terms. On the other hand, change requesters serving another customer, called in the following customer B, describe their tickets in a lengthy style more close to natural language, creating a vocabulary of 3446 terms. The dataset of customer B has approximately the same size: 1317 tickets covering 88 categories (see Figure 5b). Worth mentioning is that the tickets are not evenly distributed across the categories: some categories are over-represented, whereas others have just some instances.

A. Information Retrieval Approach

In the information retrieval approach we index for each category several of its fields (title, description, service line) as well as human-defined keywords specific for the given customer. To avoid indexing non-relevant tokens, some English stopwords (like *the* or *a*) were removed. Furthermore, specific words were expanded to their synonyms (e.g. *admin* $\rightarrow$ {*admin, administrator, SA, sysadmin*}). We use ticket information (summary and owner group's mapping to service line) to search the static index, using different hit weights depending on the field.

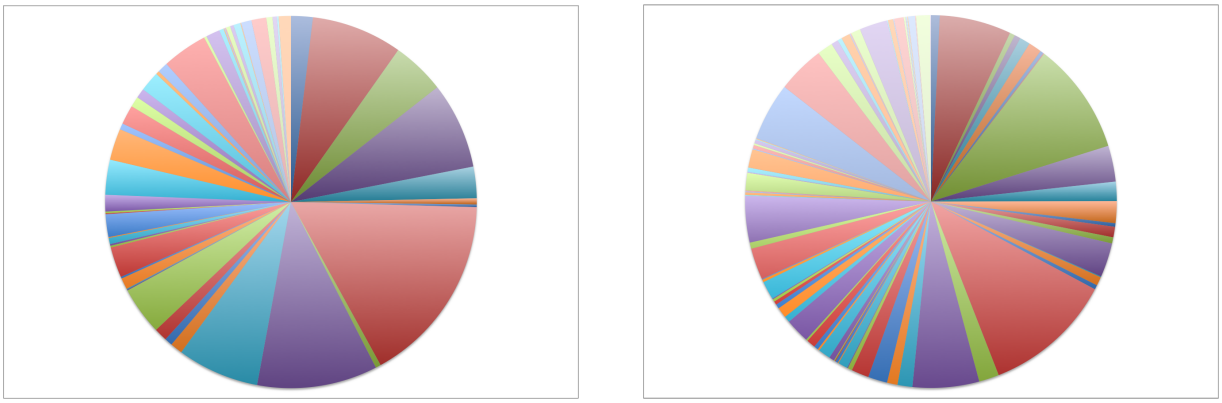
B. Machine Learning Approach

For all machine learning experiments we performed a *10-fold cross-validation*. Initially, the dataset is randomly partitioned into 10 samples. From these, in every of the 10 runs (the so-called *folds*), one sample is retained for testing, while the other 9 are used to train the classifier. The average accuracy across all runs is reported. Apart from that, learning curves were built, by feeding gradually more labeled data (instances or features) to the classifier. In the supervised setting, we start with 10% of the labeled instances and increase up to 90% – the maximum size of the training set. In the weakly-supervised setting, we increase the number of labeled features in each step by 50 or 100, until the learning curve reaches a plateau. In this way, we can determine the learning speed of the classifier and the minimal amount of training data that is actually necessary to achieve a satisfactory level of performance.

As features for the classifiers we only use words from the short summary and the owner group fields of each ticket. Examples of features in the short summary are "cirats", "backup", "publication", "mq". The owner group field simply contains the name of the expert group, which is associated with a certain service line, perfoming the change. Minimal preprocessing was applied on the data: lowercasing and removing of special characters from the summary input. As feature selection, a list of English stopwords was removed from the bag of words. Each document is represented as a vector of words and their frequency in the corpus.

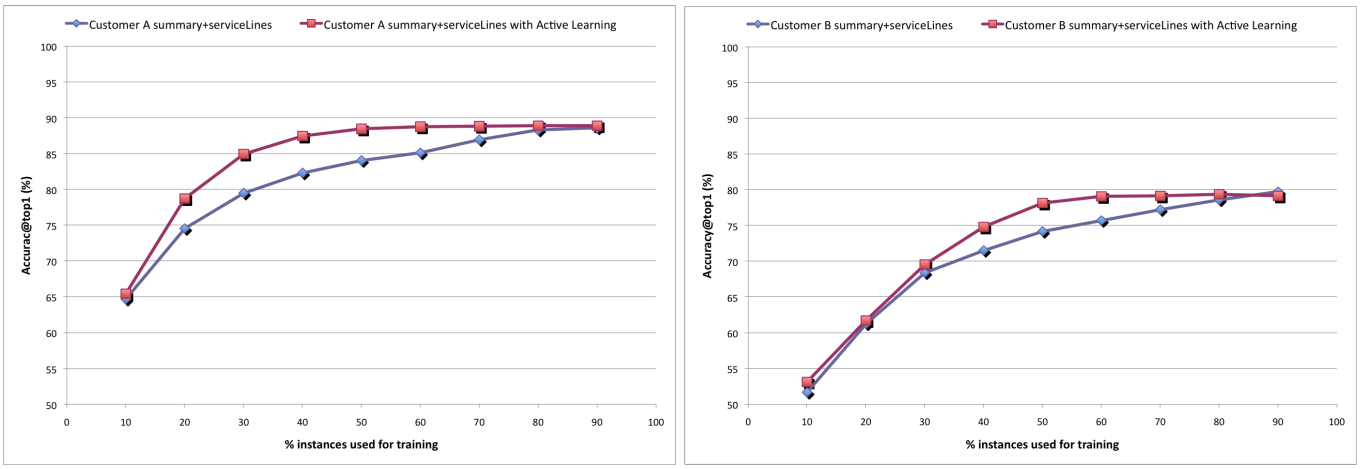
In the experiments for the supervised setting, the owner group was mapped to the corresponding service lines. These and the words in the short summary were used as features of each ticket. We tested two methods for selecting the training instances. Initially, we input to the algorithm instances randomly selected from the training set, increasing the input in several steps. In the second method, we make use of the active learning paradigm: we retrain the classifier after each step and apply it on the remaining unlabeled instances in the pool. The instances to be labeled and used for training in the next step are the instances about which the classifier is currently most uncertain how to label (based on the *margin* metric explained previously in Section IV).

In the experiments for the weakly-supervised setting, we exploit the same two fields, namely short summary and owner group. The owner group features are automatically labeled by first mapping th eowner group to the service lines it supports and then labelling each service line is automatically with the activity categories belonging to it. For labeling features from the short summary field, two methods were employed for selecting candidate features. Firstly, we use Latent Dirichlet Allocation (LDA) [21] to cluster the unlabeled data around latent topics. As this is an unsupervised technique, the discovered topics are not necessarily aligned with the user-defined topics (i.e. categories). We set the



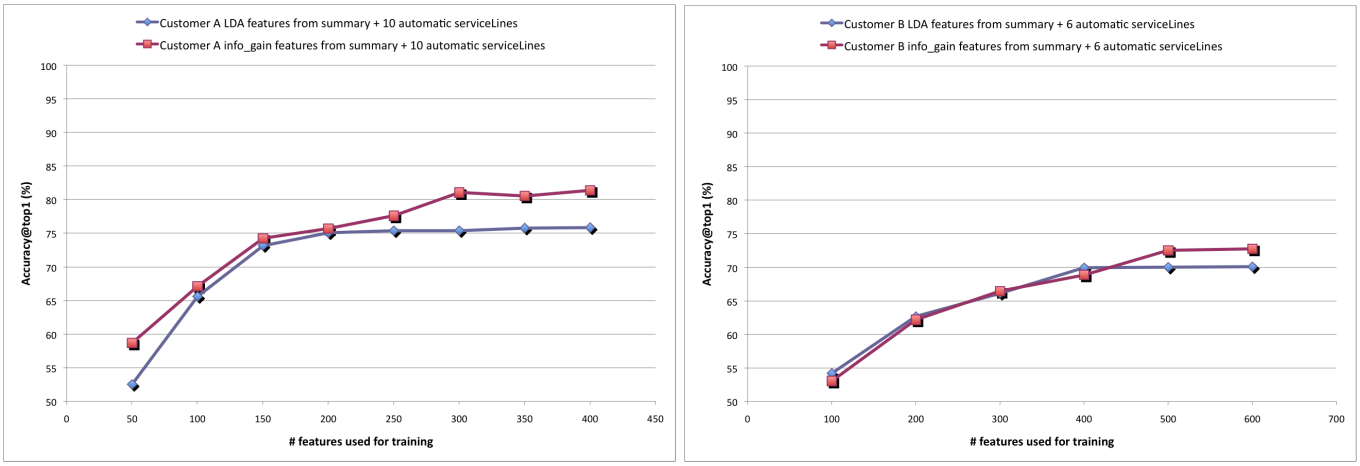
(a) Customer A (b) Customer B

Figure 5. Customers have different distributions over classes: 54 categories for customer A; 88 categories for customer B.



(a) Customer A (b) Customer B

Figure 6. Accuracy@top1 vs. percentage of instances used for training by the LR classifier.



(a) Customer A (b) Customer B

Figure 7. Accuracy@top1 vs. number of features used for training by the GE classifier.

number of topics to 50, independent of the dataset. For each of the LDA topics, the features $\mathbf{x}_j$ are sorted descending according to their predictive power for the respective topic; finally, only the top features of each cluster are returned. In this paper, we take the top $25L$ features according to these two metrics, where L is the number of classes in the data. Secondly, to obtain an upper bound on feature selection methods, we select features according to their predictive power as measured by the mutual information of the feature with the class label. The mutual information of the features within each class is computed by an oracle - a common experimental setup when human labelers are not available - making used of the true instance labels. If the mutual information is above a given threshold, the oracle labels the feature with the class under which it occurs most often, and also with any other class under which it occurs at least half as often. In the experiments we use as threshold the mean of the mutual information scores of the top $100L$ most predictive features; and $q_{maj} = 0.9$ as the majority of the probability mass to be distributed among classes associated to a labeled feature. The oracle is conservative in practice, simulating a scenario in which the user only knows about the most prominent features.

VI. Results and Discussion

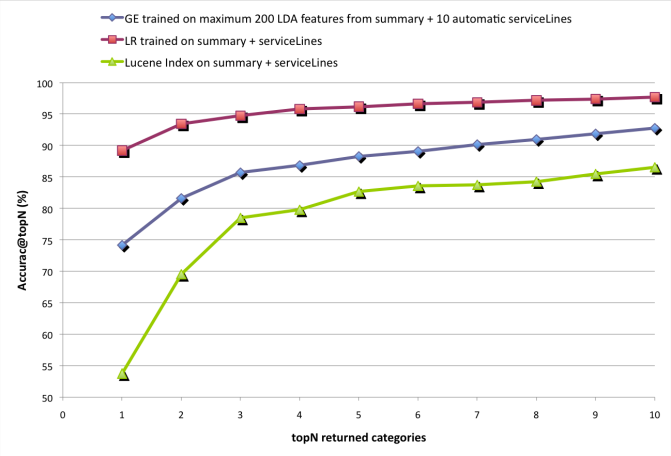


Figure 8. Comparison of the three approaches for customer A.

The results of all approaches are presented using *accuracy*, also called *success rate*, as the evaluation metric. For the binary case, the accuracy metric is defined as:

$$Accuracy = \frac{(tp + tn)}{(tp + fp + tn + fn)}$$

where *tp* are the true positives, *fp* the false positives, *tn* the true negatives, and *fn* the false negatives of the confusion matrix [22]. In general, accuracy refers to the percentage of correct predictions made by the model when compared with the actual classifications and can be computed as the sum of correct classifications (the sum of the elements in the main diagonal of the confusion matrix) divided by the total number of classifications (the sum of all elements of the confusion matrix).

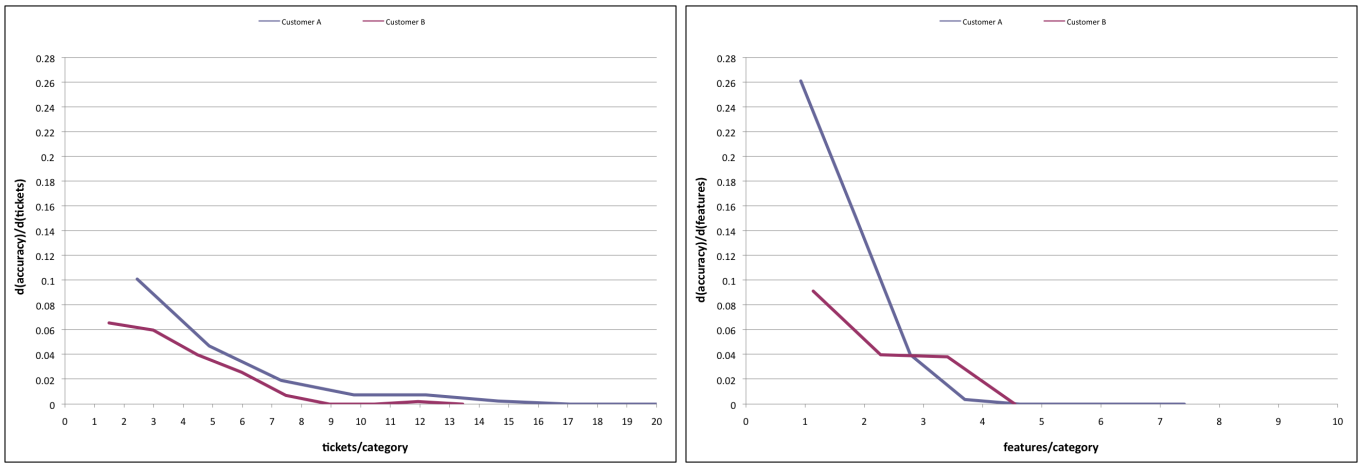
In the following, *accuracy@top1* represents the fraction of instances that have the correct label as their best predicted label, whereas *accuracy@topN* represents the fraction of instances that have the correct label within the $topN$ returned labels.

Figures 6a and 6b present the learning curves of the multinomial logistic regression classifiers with and without active learning. The results show that the active learning approach to labeling tickets invariably outperforms the baseline approach, while, in addition, greatly reduces the supervision requirements – already at 50% of the data the active learning models reach the maximum accuracy, otherwise reached by the baseline models only when training on 90% of the data. Moreover, the performance of the classifiers induced on the customer A data is significantly better than of those on customer B: 88.6% vs. 79.7% accuracy. This comes as no surprise, as firstly, the task of distinguishing between 54 categories is easier than choosing from 88 categories and secondly, the size of the vocabulary is considerably smaller. In other words, with fewer classes and features, there are less θ parameters to be estimated.

Figures 7a and 7b show the learning curves obtained by varying the number of labeled features input to the generalized expectation method. From these curves we are able to obtain a deeper insight into the supervision requirements of our weakly-supervised approaches. We conclude that, when using features selected by the topic modeling, as little as 200 features for customer A and 400 features for customer B are enough to achieve performances on the plateau of the curves. When the candidate features for labeling are selected according to their information gain, the classifiers reach better accuracies because this method, having access to the instance labels, discovers more discriminative features.

Figure 8 compares the three approaches on the customer A data. It can be observed that the *accuracy@top1* values are very different across the three approaches: the Lucene index yields an accuracy of just 53.8%, while the logistic regression classifier already achieves an accuracy of 89.2%. Starting with 5 returned categories, the performance of the classifiers does not improve significantly anymore – this is the reason why we always presents to the user the top 5 matching categories. At this point, both of the machine learning approaches reach accuracy levels of over 88%, a satisfactory threshold.

The reader must bear in mind that the GE approach is based on the new paradigm of labeling words (as opposed to labeling whole documents), which is less time consuming and more natural for domain experts. For example, on the customer A data, the LR classifier without active learning requires 265 training documents to achieve an accuracy of



(a) Learning rates vs. tickets/category needed in the training phase by the LR classifier with active learning.

(b) Learning rates vs. features/category needed in the training phase by the GE classifier.

Figure 9. Learning rates of the two machine learning approaches.

75%. In comparison, the GE classifier achieves the same accuracy level with only 200 labeled words for training.

Figure 9a presents the slopes of the learning curves for the supervised approach vs. the average number of labeled tickets per category. From these curves we are able to conclude that, an average number of 7 pre-classified tickets per category is enough to reach the maximum accuracy, independently of the the dataset. We believe that the slight translation of the curve of customer A to the right is caused by the nature of its corpus: tickets that are in the same category are actually very different and do not share common terms. On the other hand, the wordiness of the tickets from customer B generates ”overlaps” between the tickets and creates representative features for each category.

Figure 9b, on the other hand, presents the slopes of the learning curves for the weakly-supervised approach vs. the average number of labeled features per category. Taking the same learning rate as threshold, i.e. 0.02, we observe that, as little as 4 annotated features per category are enough to acquire top performances across the two customers. The cost of labeling is thus reduced when labeling features rather than instances both because fewer features need to be labelled compared to instances and because feature labeling is less costly than instance labeling [23].

We suspect these findings suggest the existence of a ”golden ratio” of the system. It is our expectation that, with a higher number of trial customers, we will be able to demonstrate that the same constant number of labeled tickets or labeled features per category is enough to achieve the plateau of the learning curves across all customers. By just knowing the number of categories a new client uses, we could then deduct the expected number of annotated data (either tickets or words) needed to train an accurate classifier. This would allow us to better estimate the effort of embarking a new client and to develop different pricing policies.

VII. Conclusions and Future Work

In this paper, we presented the application of several approaches to automatic classification of change requests that aim at decreasing the cost of training. We used a classification schema based on a catalogue of standard activities performed to service outsourced IT infrastructure, containing up to 200 fine-grained activities. The effort of labeling historical data was reduced by employing active learning techniques or by leveraging labeled features.

The first of the methods, the information retrieval approach (a Lucene index), reported the worse performance and did not scale-up with the increasing number of keywords for each customer. The supervised machine learning setup (a multinomial logistic regression classifier) yielded very good results (89.2% accuracy@top1 and 96.2% accuracy@top5 for customer A), but required large amounts of training data for each customer. To speed-up the learning process, we employed active learning techniques and reduced to half the number of required labeled documents for training. The semi-supervised setup with weakly-labeled data (a generalized expectation criteria classifier) resulted in satisfactory performance (74.2% accuracy@top1 and 88.3% accuracy@top5 for customer A) at a very moderate cost: 200 labeled words. By this means, we were able to automatically exploit the extra information we have on the class, like description or service line, and to let the domain experts/oracles point out in a less time consuming and more natural way the affinities between the input features and the classes.

There are several possible avenues for future work that we would like to explore. Firstly, we are currently exploring

ways of porting existing statistical models induced from the corpus of one customer to other customers leveraging the existing body of work on transfer learning across domains. Secondly, we will study the interplay between labeled tickets and labeled features through a set of experiments which will allow us to analyze the behavior of the induced model under varying amounts of labeled features and labeled documents. This will build and extend prior work of Raghavan [23] who has deployed a dual supervision in an active learning setup. However, while in Raghavan's work the feature labeling merely results in a fine tuning of the classifier trained with labeled instances - either by pruning the feature space or by boosting certain features, our combined objective function is able to exploit both labeled instances and features. This consists in a more principled way of combining the various types of supervision available – feature and instance labels – than previous approaches that perform model induction and feature selection in separate steps, and it is thus expected to lead to enhanced classifier accuracy.

ACKNOWLEDGMENT

The authors would like to thank the rest of the development team: Catalin-Mihai Barbu, Sinem Guven, Sinziana Mazilu, Lev Merenkov, Madhumita Sadhukhan, and Christoph Thomet.

REFERENCES

[1] A. Bose, A. Heching, and S. Sahu, "A framework for model-based continuous improvement of global it service delivery operations," *In IEEE International Conference on Services Computing*, 2008.

[2] O. of Government Commerce, *The Official Introduction to the ITIL Service Lifecycle*, 2nd ed. The Stationery Office (TSO), Norwich, 2010.

[3] G. Spalding and G. Case, *ITIL Continual Service Improvement*, 2nd ed. The Stationery Office (TSO), Norwich, 2007.

[4] EMA, "Adding control to change management: how to assess your requirements," 2007.

[5] G. A. D. Lucca, M. D. Penta, and S. Gradara, "An approach to classify software maintenance requests," *In ICSM*, 2002.

[6] R. Gupta, K. H. Prasad, L. Luan, D. Rosu, and C. Ward, "Multi-dimensional knowledge integration for efficient incident management in a services cloud," *In IEEE International Conference on Services Computing*, 2009.

[7] Y. Diao, H. Jamjoom, and D. Loewenstern, "Rule-based problem classification in it service management," *In IEEE International Conference on Cloud Computing*, 2009.

[8] M. McCandless, E. Hatcher, and O. Gospodnetic, *Lucene in Action*, 2nd ed. Manning Publications, 2010.

[9] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of Documentation*, vol. 28, pp. 11–21, 1972.

[10] F. Sebastiani, "Machine learning in automated text categorization," *ACM Computer Survey*, vol. 34, pp. 1–47, 2002.

[11] G. Forman, "An extensive empirical study of feature selection metrics for text classification," *Journal of Machine Learning Research*, vol. 3, pp. 1289–1305, 2003.

[12] T. Zhang and F. J. Oles, "Text categorization based on regularized linear classification methods," *Information Retrieval*, vol. 4, pp. 5–31, 2001.

[13] K. Nigam, J. Lafferty, and A. Mccallum, "Using maximum entropy for text classification," *In IJCAI-Workshop on Machine Learning for Information Filtering*, 1999.

[14] B. Settles, "Active learning literature survey," University of Wisconsin–Madison, Tech. Rep. 1648, 2010.

[15] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," *IN SIGIR*, 1994.

[16] T. Scheffer, C. Decomain, and S. Wrobel, "Active hidden markov models for information extraction," *In CAIDA*, 2001.

[17] G. Mann and A. McCallum, "Generalized expectation criteria for semi-supervised learning with weakly labeled data," *Journal of Machine Learning Research*, vol. 11, pp. 955–984, 2010.

[18] G. Druck, G. Mann, and A. Mccallum, "Learning from labeled features using generalized expectation criteria," *In SIGIR*, 2008.

[19] S. Kullback and R. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, vol. 22, pp. 79–86, 1951.

[20] R. E. Schapire, M. Rochery, M. Rahim, and N. Gupta, "Incorporating prior knowledge into boosting," *In ICML*, 2002.

[21] D. M. Blei, A. Y. Ng, M. I. Jordan, and J. Lafferty, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[22] R. Kohavi and F. Provost, "Glossary of terms," *Editorial for the Special Issue on Applications of Machine Learning and the Knowledge Discovery Process*, vol. 30, 1998.

[23] H. Raghavan, O. Madani, and R. Jones, "Active learning with feedback on both features and instances," *In Journal of Machine Learning Research*, vol. 7, pp. 1655–1686, 2006.