2014 IEEE International Conference on Data Mining

Hierarchical Incident Ticket Classification with Minimal Supervision

Andrii Maksai*, Jasmina Bogojeska*, and Dorothea Wiesmann*
*IBM Research - Zurich, Säumerstrasse 4, CH-8803 Rüschlikon, Switzerland, {aks, jbo, dor}@zurich.ibm.com

***Abstract*—In this paper, we introduce a novel approach for incident ticket classification that aims at minimizing the manual labelling effort while achieving good-quality predictions. To accomplish this, we devise a two-stage technique that employs hierarchical clustering using a combination of graph clustering (community finding) and topic modelling as first stage, followed by either another round of hierarchical clustering or an active learning approach as second stage. We evaluate the performance of our method in terms of manual labelling effort, prediction quality and efficiency on three real-world datasets and demonstrate that classical approaches to text classification are not well suited for incident ticket texts.**

I. Introduction

IT service interruptions, server and applications failures, as well as any issue across server stacks are governed by the Information Technology Infrastructure Library (ITIL) incident management process and documented in incident tickets [1]. The main goal of incident management is to minimize the business impact of incidents by restoring normal service operation as quickly as possible. The documentation in the incident tickets contains a symptom description of the issues as well as details on the incident resolution and a range of structure fields, e.g., date, resolver, servers and services affected. A classification of the incident tickets that occurred in a specific IT environment will thus yield a good overview on the types of issues present. This will enable efficient overall incident management by focusing on the problems of the most critical or most pervasive incident types and by determining actions to prevent them from happening in the future. For example, IT service managers could focus specifically on servers with server unavailability incidents and correct the underlying issues. Server unavailability is a very important incident type as it translates into unavailable services and applications, which negatively impact business revenue and productivity. Another focus area could be nonactionable incident tickets. These are incident tickets created by monitoring systems that require no remediating actions because the situations are acceptable or transient. Although these incident tickets do not represent service interruptions, they divert attention from critical situations. Thus identifying and remediating these, e.g., by fine-tuning monitoring systems [2], will increase productivity and focus on critical situations.

In this paper, we present a novel method for incident ticket classification. This is a very challenging task for the following reasons:

- The number of tickets is very large (on the order of thousands per year for a large IT environment), which makes their manual labelling practically impossible. Also, different ticket types have very different sample abundances.
- Ticket resolution is a mixture of human and machine generated text (from the monitoring system) with a very problem-specific vocabulary.
- The texts of the tickets from different IT environments are very different as they are written by different teams who use different monitoring systems, which renders reuse of manually labelled tickets and knowledge transfer among different IT environments infeasible.

Our method consists of two steps. In the first step, tickets are grouped using two-level hierarchical clustering. Afterwards, each cluster is first labelled using appropriately chosen sample tickets, and then clusters that have tickets of the same type are joined together for the second step, which comes in two variations: the first one applies another round of hierarchical clustering to produce the final ticket classification, whereas the second one uses active learning. The main goal of our approach is to reduce the manual ticket labelling effort while maintaining good prediction quality. This enables the system-administrator teams to do the ticket classification themselves. As they are the experts with the deepest insight into the target environment, this ensures accurate ticket labelling, which is important for the quality of the final classification.

II. Related Work

Various supervised machine-learning techniques, such as support vector machines [3], [4], [5], Naïve Bayes [6], [7], maximum entropy (multinomial logistic regression) [8] and gradient boosting machine (GBM) [9], [10], have been proposed for automatic text classification. Some of these methods as well as rule-based approaches have been applied for maintenance and incident ticket classification [11], [12], [13], [14], [15].

An important area in text mining is topic modelling, in which each document is assumed to be a mixture of abstract topics. In this way, a set of documents can be clustered based on their most prominent topics using an unsupervised approach. A number of recent works in this area are based on Latent Dirichlet Allocation (LDA) [16]. For example, in [17] a procedure for obtaining a hierarchical topic structure is described, and [18] introduces a background topic for detecting uninformative words that occur across all topics.

A different approach to text classification is building a document-document graph and then clustering the documents by finding communities in the graph [19].

Finally, reducing the labelling effort for various classification tasks is an important problem addressed in many works. Active learning achieves this by automatically selecting the most informative unlabelled samples for manual annotation [20].

III. Incident Ticket Classification

In this paper, we focus on the incident ticket classification problem, i.e., the problem of identifying the type of each server incident ticket. Formally, incident ticket classification is a multi-class classification problem that aims at assigning the correct type to each ticket based on input features extracted from the ticket resolution text. We did not use the symptom description as most of its information was already contained in the resolution text. Although we consider five incident types, namely, *server unavailable (SU)* (unexpected shutdown, reboot, defect hardware, system hanging), *disk (D)* (disk capacity problems), *process-app (PA)* (missing processes, virus issues, application problems), *nonactionable (NA)* (no action taken) and *other (O)* (all remaining incidents), different classes can be specified depending on the set of incident tickets and their origin.

A. Method Description

Our approach tackles the problem of incident ticket classification while keeping the manual labelling effort minimal. In this way, as the ticket annotation needed requires only little time, it can be done by subject-matter experts. Note that achieving good quality predictions is as important as minimizing the manual labelling effort. Algorithm 1 gives a very general overview of our ticket classification approach, and in what follows, we provide the details of each step.

Algorithm 1 Hierarchical Incident Tickets Classification (HITC)

```
Input: Set of incident tickets with available resolution texts;
set $option$ for the second stage either $hierarchical$ for two-
level hierarchical clustering or $active$ for active learning.
  1) Extract bag-of-words input features from the incident
     ticket resolution texts;
  2) Cluster the tickets using 2-level hierarchical clustering;
  3) Assign multi-type labels to each ticket group;
  4) For each ticket type $t$ do:
       • Merge all clusters from step 2 that contain ticket
         type $t$;
       • if $option == hieararchical$ apply hierarchical
         clustering on the merged data and assign a ticket
         type to each cluster;
         if $option == active$ use active learning on the
         merged data to discriminate between the target ticket
         type $t$ and the remaining types.
```

B. Hierarchical Incident Ticket Clustering

As a first step towards solving the ticket classification problem, we devise a two-level hierarchical clustering procedure as follows. First we build a ticket-term graph in which each ticket and each term in the word corpus are represented by a node, and there is an edge between a ticket node and a term node if the target resolution text contains the respective term. First-level clusters are then obtained using graph clustering, which discovers dense structures (communities) in the ticket-term graph. For this purpose, we use the community detection algorithm introduced in [21], optimized for large networks in [19] and referred to as Clauset–Newman–Moore algorithm. As the first-level clustering is done for the complete (large) dataset, this ensures efficient computation. Note that the number of clusters is chosen automatically using the notion of modularity. It quantifies the quality of the graph clustering and it is higher when there are few edges between communities and many within them. Intuitively, applying graph clustering on the ticket-term graph groups the tickets that share many common words in their corresponding resolution texts. To obtain second-level clusters, first Latent Dirichlet Allocation (LDA) [16] is applied on each cluster from the previous (first) level. LDA is a standard topic modelling approach in which each ticket is represented as a mixture of a small number of ticket types. After running LDA, the second-level clusters are obtained by grouping the tickets with the highest probability for each of the k topics.

In the final step of the hierarchical ticket clustering procedure, we need to assign ticket types to the second-level clusters. We do this as follows: Given a cluster, we first run LDA with a large number of topics (e.g. $k_2 = 10$). The intuition behind the large k_2 is that we want to capture all ticket types considered. Second, for each of the generated clusters (topics) that are large enough (e.g., comprise more than $p = 1\%$ of the tickets), we select the ticket with the highest probability for the target topic as representative ticket for manual labelling. Finally, we annotate each cluster with the set of ticket types of the manually labelled representative tickets of its large subclusters. This approach can easily be adapted to assign a single label to each cluster by selecting a ticket only from the largest subcluster.

Finally, we briefly justify the choice of a hierarchical approach and multi-type labels. The resolution texts of tickets from one type can be very similar to the resolution texts of other ticket types. For example, the resolution text describing a real server unavailable problem (ticket type *Server Unavailable*) and that of a reported server unavailability that was a false alert (ticket type *Nonactionable*) differ only in a couple of words (e.g. “false alarm”, “false alert”, “everything fine”). Therefore, a hierarchical approach yields better ticket clustering in the lower levels of the hierarchy than a single-level clustering does. Furthermore, as the goal is to label as few tickets as possible, we build a two-level hierarchy (which yields non-homogeneous ticket clusters) and allow multi-type cluster labels. In this way when we merge the clusters for a certain ticket type, we capture almost all such tickets and can then refine the ticket classification in the final step.

```
Algorithm 2 Hierarchical Incident Ticket Clustering
Input: Input features derived from the dataset of ticket resolutions $D = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$; number of topics $k$ for the second-level clustering; percentage of samples $p$ that indicates a large cluster.
  1) Generate the first-level ticket clusters $\mathbf{C_1}$ by applying the Clauset–Newman–Moore community detection (graph clustering) algorithm on the ticket-term graph;
  2) For each cluster $c_1 \in \mathbf{C_1}$ do:
       • Use topic modelling (LDA) with $k$ topics to generate the second-level clusters $\mathbf{C_{12}}$ for $c_1$;
       • Assign multi-type labels to each second-level cluster $c_{12} \in \mathbf{C_{12}}$:
           – Cluster the tickets in $c_{12}$ by running LDA with a large number of topics;
           – For each large cluster $\mathbf{C}$ ($|\mathbf{C}| \geq p \cdot N$) from the preceding step, choose the ticket with the highest cluster (topic) probability as representative ticket for manual labelling;
           – The multi-type label for cluster $c_{12}$ is given by the set of manually annotated tickets from the preceding step.
```

C. Final Ticket Classification

One of the main goals of our approach is to achieve good-quality incident ticket classification with minimal supervision, i.e., with as few manually labelled tickets as possible. To achieve this, we provide two options for the final ticket classification: the first one applies a hierarchical ticket clustering for each ticket type and assigns a ticket class to each cluster, and the second one performs the final ticket classification by using an active learning approach for each ticket type.

1) Hierarchical Clustering for Ticket Classification: The final ticket classification can be done using a modified version of the hierarchical incident ticket clustering described in Algorithm 2. We now use three-level clustering, and obtain each of them using topic modelling (LDA). In this setting, we can change the amount of manual labelling effort by varying the number of clusters (topics). Finally, the label of each cluster is determined by the type of the ticket with the highest topic (cluster) probability. The details of this procedure are given in Algorithm 3.

2) Active Learning for Ticket Classification: Active learning is an iterative learning algorithm able to automatically select the most informative unlabelled samples for manual annotation [20]. We use a confidence-based stopping criterion with an update strategy for the threshold as described in [22]. More specifically, whenever the overall uncertainty measure given by the average cross-entropy of the unlabelled tickets is below the current threshold and the ticket labels of the current iteration have changed compared to the preceding one, the threshold is updated (halved). As we want to keep the number of manually annotated tickets minimal, we choose only ten tickets for manual annotation in each iteration of the active learning process and also set an upper limit on the number of manually labelled tickets. Similar to Algorithm 3, the active learning approach first merges the data from all clusters annotated by ticket type t produced by Algorithm 2. Then, label 1 is assigned to all manually labelled tickets (set $\mathbf{MLS} = \{(x_i, t_i), i = 1, \ldots, l\}$) from type t and label -1 to the rest, and $\mathbf{LS} = \{(x_i, y_i) | i = 1, \ldots, l, y_i \in 1, -1\}$ is used as initial labelled set. Finally, the active learning procedure from [22] is applied with a logistic regression model as a learning method employed in each iteration.

```
Algorithm 3 Hierarchical Clustering for Ticket Classification
Input: Target ticket type $t$; all clusters annotated by $t$ produced by Algorithm 2; number of topics $k_1$, $k_2$, $k_3$ for first-, second- and third-level clustering, respectively.
  1) Merge the data from all clusters produced by Algorithm 2 annotated by the ticket type $t$;
  2) Generate the first-level ticket clusters $\mathbf{C_1}$ using topic modelling (LDA) with $k_1$ topics;
  3) For each cluster $c_1 \in \mathbf{C_1}$ do:
       • Run LDA with $k_2$ topics to generate the second-level clusters $\mathbf{C_{12}}$ for $c_1$;
  4) For each cluster $c_{12} \in \mathbf{C_{12}}$ do:
       • Run LDA with $k_3$ topics to generate the third-level clusters $\mathbf{C_{123}}$ for $c_{12}$;
       • Assign a ticket type to each third-level cluster $c_{123} \in \mathbf{C_{123}}$:
           – Select the ticket with the highest cluster probability as representative ticket for manual labelling;
           – The label for cluster $c_{123}$ is given by the type of the manually annotated ticket from the preceding step.
```

Finally, note that because in both approaches we are merging multi-type clusters for the final ticket classification, there can be tickets annotated with more than one label. In such cases, we choose the label with the highest class probability – these are either probabilities estimated in the active learning process or topic probabilities estimated with LDA.

IV. Experimental Evaluation

In this section, we evaluate the performance of the HITC-based methods in terms of ticket classification quality and manual annotation effort. We do this by comparing the performance of the HITC methods with related approaches for text classification.

A. Datasets

We use real-world incident ticket datasets from three IT environments (ITEs): 24517 tickets in total and 1718 manually labelled tickets from ITE1, 24473 tickets in total and 988 manually labelled tickets from ITE2, and 9608 tickets in total

TABLE I: Ticket class abundances for the different ITEs.

Dataset	Server unavailable	Disk	Process-App	Nonactionable	Other
ITE1	98 (5.7%)	194 (11.29%)	0 (0%)	1247 (72.58%)	175 (10.38%)
ITE2	96 (9.71%)	109 (11.03%)	74 (7.48%)	635 (64.27%)	74 (7.48%)
ITE3	83 (4.37%)	363 (19.15%)	0 (0%)	1120 (59.1%)	329 (17.36%)

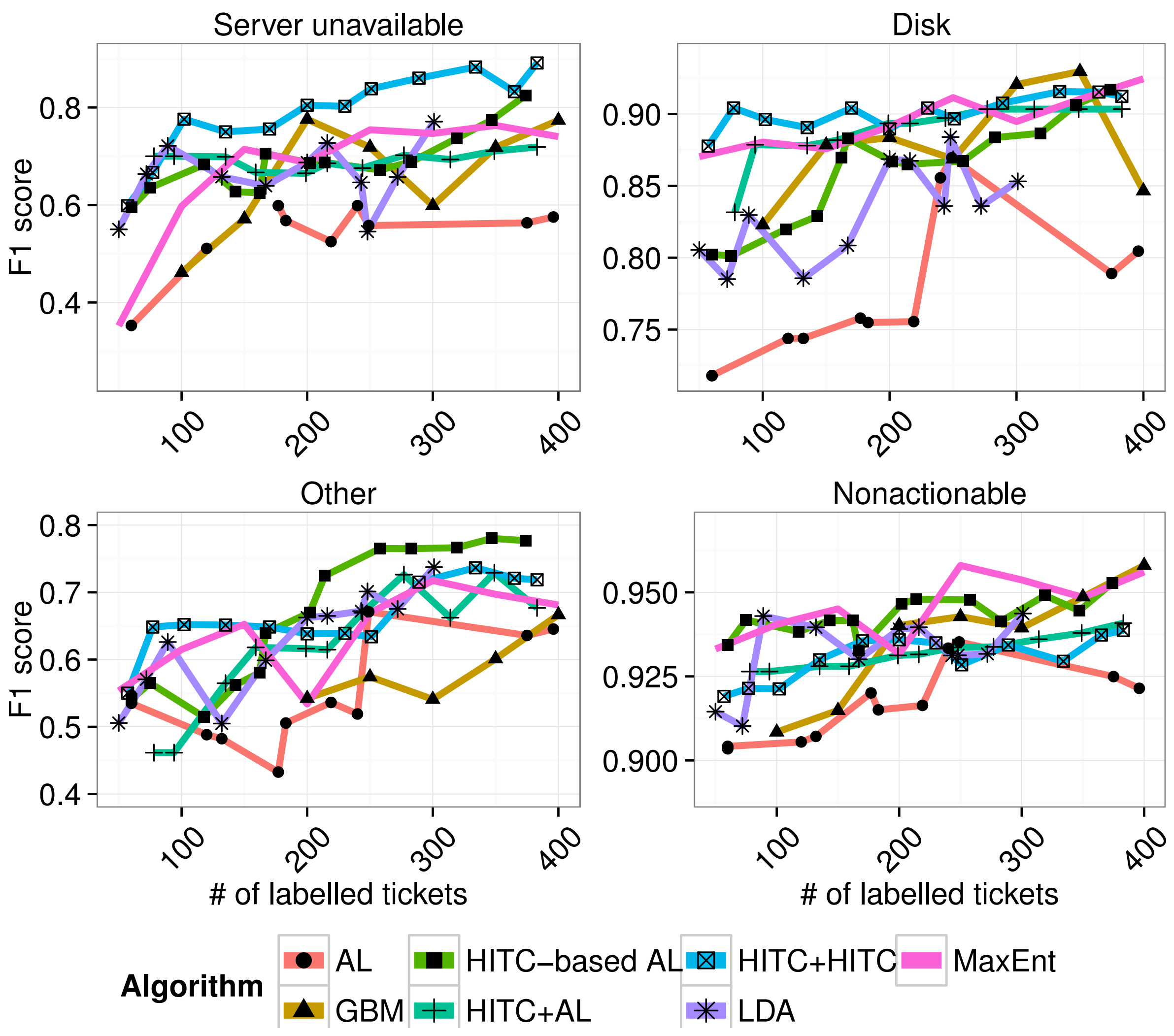

Fig. 1: F-scores of all methods considered based on the number of tickets labelled for the ticket classes in ITE1.

and 1895 manually annotated tickets from ITE3. The ticket types with their corresponding counts (based on the labelled sets) for each ITE are given in Table I, where it can be seen that the different ticket types have very different sample abundances.

B. Comparison Methods

In the following, we briefly describe all the methods we consider for comparison when evaluating the performance of our HITC methods. We will refer to the HITC method that uses hierarchical clustering for the final ticket classification as *HITC+HITC* and to the one that uses active learning as *HITC+AL*.

Maximum Entropy (MaxEnt) Model. It is a generalization of the logistic regression model to multi-class problems [23] and is often applied to text classification tasks.

Active Learning with MaxEnt as Base Classifier. We implement the active learning approach using randomly chosen labelled initialization set, the MaxEnt model as base classifier, and the overall uncertainty as a stopping criteria as described in [22]. This method is referred to as *AL*. We also consider a variant of this active learning approach, denoted as *HITC-based AL*, in which we replace the randomly chosen initialization set by the set of tickets annotated in the first stage of

TABLE II: Average F-Scores±Standard Deviation (first row in each cell) and Precision, Recall (second row in each cell) obtained for all methods on all ticket types for each IT environment in the bootstrap analysis. All approaches use $100-150$ labelled tickets. We also provide the sample abundance for each ticket type. Best results are marked in bold.

Algorithm	ITE1 SU (5.7%)	ITE1 NA (72.5%)	ITE1 D (11.2%)	ITE1 O (10.3%)	ITE2 SU (9.7%)	ITE2 NA (64.2%)	ITE2 D (11.0%)	ITE2 O (7.5%)	ITE2 PA (7.5%)	ITE3 SU (4.37%)	ITE3 D (19.2%)	ITE3 NA (59.1%)	ITE3 O (17.4%)
HITC+HITC	**71.2±1.7** 73.7 69.5	**93.8±1.2** 96.7 91.2	**89.3±2.9** 99.2 81.4	**73.4±2.1** 96.0 59.9	**74.2±2.1** 94.1 61.7	81.4±0.8 91.9 73.1	**61.1±1.6** 93.6 46.3	48.1±1.5 96.5 32.6	**44.3±2.2** 93.6 30.0	**37.9±1.8** 38.0 38.7	61.1±2.0 95.3 46.3	82.3±0.8 88.7 76.9	**74.5±1.8** 93.0 62.6
HITC+AL	66.9±2.7 71.8 64.2	89.7±2.6 89.8 89.9	84.0±1.2 84.4 84.0	53.4±1.1 53.7 53.7	64.5±2.4 73.7 59.6	76.9±1.9 87.9 70.1	50.9±2.5 98.9 34.4	39.3±2.9 63.1 30.9	38.1±1.3 56.6 30.9	29.9±1.0 30.8 29.8	49.5±1.7 59.7 43.0	82.1±2.7 93.6 76.6	62.6±2.5 71.6 56.7
HITC-based AL	64.5±1.3 81.1 57.4	91.3±2.0 88.1 94.8	81.1±2.3 91.0 74.0	44.1±1.8 79.7 32.6	65.9±1.5 68.9 69.9	**83.8±1.6** 84.7 83.9	53.6±2.2 73.3 45.8	43.8±1.8 67.9 35.2	34.5±2.2 59.6 26.7	28.7±2.7 56.9 21.9	46.9±1.1 59.7 45.0	80.7±1.9 76.2 86.2	73.1±1.8 89.2 64.4
Active Learning	45.7±2.9 66.8 40.9	90.5±1.4 88.0 93.3	74.2±2.3 91.1 64.5	38.2±1.5 56.4 31.4	56.5±1.6 73.5 52.1	76.1±2.4 73.6 79.5	41.3±1.5 48.4 39.4	30.2±1.5 62.0 22.4	43.2±1.1 80.6 30.6	10.4±2.8 53.4 7.1	53.4±1.2 58.8 52.1	77.9±1.7 72.4 84.7	48.3±2.3 81.8 36.3
GBM	65.0±1.3 72.4 61.3	93.1±1.3 90.3 96.2	86.3±1.4 93.6 80.6	50.1±1.5 67.6 42.5	47.4±2.1 49.1 47.1	78.5±0.5 66.7 95.8	32.1±1.6 42.6 28.7	5.1±2.9 7.5 6.0	8.3±1.8 15.2 8.4	19.7±2.1 37.0 16.3	57.3±3.0 65.6 53.2	78.5±1.2 75.6 82.2	43.9±1.7 46.7 47.1
MaxEnt	65.6±2.3 72.9 63.6	93.3±1.2 92.3 94.5	87.1±1.4 89.6 85.4	58.4±1.0 65.8 54.8	53.6±2.2 52.2 57.0	78.1±2.8 74.8 82.1	59.7±1.4 69.3 55.0	44.5±1.2 59.8 37.1	32.8±2.3 41.4 30.3	31.2±2.7 34.4 42.1	**63.0±2.4** 64.7 62.8	**86.2±2.3** 84.2 88.7	74.3±1.7 79.5 70.7
LDA	62.5±1.2 62.3 63.2	90.5±1.5 94.9 87.1	74.5±1.9 87.0 66.6	52.8±2.0 58.1 50.8	50.7±1.8 52.2 50.0	77.9±1.0 96.2 66.2	53.6±2.3 55.3 53.2	**48.2±1.8** 48.9 48.7	42.5±1.6 45.5 41.1	35.8±1.3 37.6 37.3	52.8±2.1 53.0 54.0	77.4±2.7 71.4 85.1	73.3±2.1 87.3 65.4

our method (Algorithm 2).

Ticket Classification using Gradient Boosting Machine (GBM). We implement the method for ticket classification as described in [15]. It creates a representative set for manual labelling using k-means clustering to first group the tickets into clusters with similar texts and then randomly choose a number of samples from each cluster such that more samples are selected from less homogeneous clusters. Once a labelled training set is available, it is used to train a GBM model (ensemble of decision trees) [10] for multi-class ticket classification.

Latent Dirichlet Allocation (LDA). This is a standard topic modelling technique in which each text document is a mixture of several topics and each topic has a specific word distribution [16]. We can cluster the incident tickets with LDA by using the topic probabilities for each ticket. Each cluster is then annotated using the manual label of the ticket with the highest topic (cluster) probability.

We apply the four comparison methods described above to the incident ticket classification task using different numbers of manually annotated tickets.

C. Experimental Design and Results

In the following we compare the performance of the HITC approach with the considered comparison methods in terms of prediction quality and manual labelling effort. The prediction quality is measured using the precision, recall and F-score. Our approach for incident ticket classification comprises several parameters listed in Algorithms 2 and 3. We select their values based on the method's performance on the datasets considered. More specifically, we set $k = 3$ and $p = 1\%$ in Algorithm 2 and $k_1 = k_2 = k_3 = 3$ in Algorithm 3.

First of all, we vary the total number of manually labelled tickets from 50 to 400 and report the F-Score for each ticket type for all methods considered. Figure 1 summarizes the results for ITE1. The results for the other two datasets exhibit similar trends. Recall that the ticket datasets have a very unbalanced class representation (see Table I). As one can see, the performance of the methods differs based on the size of the target class. One can observe that our HITC+HITC approach outperforms all other methods for the smallest ticket class, Server unavailable, represented with only 5.7% of the data. The other HITC-based methods are slightly worse than the HITC+HITC for this ticket class, but have comparable performance to that of the best comparison methods. Considering the classes Disk and Other, represented with 11% and 10%, respectively, HITC+HITC exhibits the best performance for lower labelling efforts (below 150 tickets) and comparable performance to the best comparison methods for higher annotation efforts. For the class Other, the HITC-based AL approach achieves the best results for labelling efforts above 200 tickets. Nonactionable is the most abundant ticket class with 72%. Therefore, most of the methods considered achieve an F-Score above 0.92 for this ticket type. While the HITC+AL and HITC+HITC approaches exhibit a slightly $(1-2\%)$ worse performance for the nonactionable type than LDA and MaxEnt, the HITC-based AL approach achieves comparable performance. GBM also achieves very good performance when the number of annotated tickets is larger than 200. Note also that the performance of HITC-based AL is much better than that of the randomly initialized AL for all ticket classes. This demonstrates the ability of the first-stage hierarchical clustering to split the different ticket types into separate clusters and to choose a proper representative ticket for each of them. Considering all HITC-motivated approaches, i.e., HITC+HITC, HITC+AL and HITC-based AL, HITC+HITC is the best alternative as it has the best performance on the smallest class (SU), the best performance for low annotation efforts for the two medium-sized classes (D and O), and a very good performance (within 2% of the best performing method with F-scores above 0.92) on the largest class (NA).

One of the main goals of the work we have presented here is to reduce the manual labelling effort as much as possible while ensuring good quality performance. Note also that by increasing the number of manually labelled tickets, we decrease the size of the test (unseen) set on which the results are reported, i.e., we increase the percentage of labelled tickets. Therefore in the following, we first fix the labelling effort between 100 and 150 tickets, which requires about 30 min to label. Then, we report the best performance of all methods considered in this range of labelling effort on all ticket classes for all available datasets (ITE1, ITE2 and ITE3). A bootstrap method [24] with 200 resamples (generated by random sampling with replacement from the original labelled data) is used for a more detailed analysis of the variability of the performance measures reported. The F-scores and their standard deviations, as well as the precision and recall values for all algorithms on all datasets are shown in Table II. For larger classes, such as Nonactionable and Disk, MaxEnt and also GBM perform very well, but fail to detect the small classes. Our HITC+HITC method has very good performance for both small and large ticket classes, achieving the highest F-score in the majority of the classes for each dataset (9 out of 13), and showing a similar performance as the best methods for the rest. HITC-based AL and HITC+AL approaches are somewhere in between, with better performance than the comparison methods for the smaller ticket classes, although worse than HITC+HITC. Traditional active learning with random initialization is not well suited for the ticket classification task.

V. Conclusions

In this paper, we presented a novel, two-stage approach that tackles the problem of incident ticket classification. In the first stage, it uses hierarchical clustering to produce multi-type clusters that capture the majority of each ticket type in their corresponding clusters. The second stage aims at refining the ticket classification for each ticket class separately by either using another round of hierarchical clustering or applying active learning. Incident ticket classification is a very challenging task, mainly because of the very specific nature of the ticket resolution texts, the different sample abundances of the different ticket types, the large number of tickets, and the large differences among tickets originating from different IT environments. The main advantage of our approaches is that they achieve good prediction quality on the ticket classification task with minimal labelling effort. This provides a fast ticket classification procedure and enables the maintenance teams, i.e., the experts from the environment in which the incidents were generated, to do the classification themselves. We evaluate the performance of our methods, and compare them with other text classification approaches on three real-world datasets. Our methods are especially powerful for very small classes, such as Server unavailable, which is an important ticket category as it translates into unavailable services and has a direct impact on business revenue.

References

[1] R. A. Randy A. Steinberg, *ITIL Service Operation*. The Stationery Office, 2011.

[2] L. Tang, T. Li, F. Pinel, L. Shwartz, and G. Grabarnik, "Optimizing system monitoring configurations for non-actionable alerts," *Proc. of the IFIP/IEEE Network Operations and Management Symposium (NOMS) 2012*, pp. 34–42, 2012.

[3] Z. Wang, X. Sun, D. Zhang, and X. Li, "An optimal SVM-based text classification algorithm," *Proc. of the 5th IEEE International Conference on Machine Learning and Cybernetics*, pp. 1378–1381, 2006.

[4] B. Zhang, J. Su, and X. Xu, "A class-incremental learning method for multi-class support vector machines in text classification," *Proc. of the 5th IEEE International Conference on Machine Learning and Cybernetics*, pp. 2581–2585, 2006.

[5] B. Rujiang and L. Junhua, "A novel conception based text classification method," *Proc. of the IEEE International e-Conference on Advanced Science and Technology*, pp. 30–34, 2009.

[6] S. Kim, K. Han, H. Rim, and S. H. Myaeng, "Some effective techniques for naïve Bayes text classification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, pp. 1457–1466, 2006.

[7] M. J. Meena and K. R. Chandran, "Naïve Bayes text classification with positive features selected by statistical method," *Proc. of the IEEE International Conference on Advanced Computing*, pp. 28–33, 2009.

[8] K. Nigam, J. Lafferty, and A. McCallum, "Using maximum entropy for text classification," *IJCAI-99 Workshop on Machine Learning for Information Filtering*, pp. 61–67, 1999.

[9] J. Friedman, "Greedy Function Approximation: A Gradient Boosting Machine," *Annals of Statistics*, vol. 29, pp. 1189–1232, 2000.

[10] ——, "Stochastic Gradient Boosting," *Computational Statistics and Data Analysis*, vol. 38, pp. 367–378, 2002.

[11] G. A. D. Lucca, M. D. Penta, and S. Gradara, "An approach to classify software maintenance requests," *Proc. of IEEE ICSM*, pp. 93–102, 2002.

[12] R. Gupta, H. Prasad, L. Luan, D. Rosu, and C. Ward, "Multi-dimensional Knowledge Integration for Efficient Incident Management in a Services Cloud," *Proc. of IEEE SCC*, pp. 57–64, 2009.

[13] Y. Diao, H. Jamjoom, and D. Loewenstern, "Rule-based Problem Classification in IT Service Management," *Proc. of IEEE CLOUD*, pp. 221–228, 2009.

[14] C. Kadar, D. Wiesmann, J. Iria, D. Husemann, and M. Lucic, "Automatic classification of change requests for improved IT service quality," *Proc. of SRII Global Conference*, pp. 430–439, 2011.

[15] J. Bogojeska, I. Giurgiu, D. Lanyi, G. Stark, and D. Wiesmann, "Impact of HW and OS Type and Currency on Server Availability Derived From Problem Ticket Analysis," *Proc. of the IFIP/IEEE Network Operations and Management Symposium (NOMS) 2014*, pp. 34–42, 2014.

[16] D. M. Blei, Y. Ng, and M. I. Jordan, "Latent Dirichlet Allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[17] D. M. Blei, T. L. Griffiths, and M. I. Jordan, "The nested Chinese restaurant process and Bayesian nonparametric inference of topic hierarchies," *Journal of the ACM (JACM)*, vol. 57, no. 2, p. 7, 2010.

[18] C. Chemudugunta, P. Smyth, and M. Steyvers, "Modeling General and Specific Aspects of Documents with a Probabilistic Topic Model," *Proc. of the Twentieth Annual Conference on Neural Information Processing Systems (NIPS)*, pp. 241–248, 2006.

[19] A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," *Physical Review E*, vol. 70, 2004.

[20] D. A. Cohn, L. Atlas, and R. E. Ladner, "Improving generalization with active learning," *Machine Learning*, vol. 15, pp. 201–221, 1994.

[21] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E*, vol. 69, 2004.

[22] J. Zhu, H. Wang, E. Hovy, and M. Ma, "Confidence-Based Stopping Criteria for Active Learning for Data Annotation," *ACM Transactions on Speech and Language Processing*, vol. 6, 2010.

[23] A. L. Berger, S. A. Della Pietra, and V. J. Della Pietra, "A maximum entropy approach to natural language processing." *Computational Linguistics*, vol. 22, pp. 39–71, 1996.

[24] B. Efron and R. Tibshirani, *An Introduction to the Bootstrap (Monographs on Statistics and Applied Probability)*. Springer, 1993.