Predicting Subcontractor Performance From Past Audits

Mertens Laurent
EAVISE
KU Leuven
Sint-Katelijne-Waver, Belgium
laurent.mertens@kuleuven.be

Vennekens Joost
EAVISE
KU Leuven
Sint-Katelijne-Waver, Belgium
joost.vennekens@kuleuven.be

***Abstract*—Utility companies, such as electricity suppliers or telecom operators, typically have to maintain large-scale infrastructure. They typically contract out maintenance work to a large collection of local subcontractors, while nevertheless retaining final responsibility. To track the quality of the performed work, the parent company dispatches its own inspectors to the sites to perform audits, particularly concerning safety and signaling. Since at any given time hundreds of sites can be open, this requires a lot of effort. In this paper, we investigate how to optimize this effort by focusing audits on those sites that are most likely to exhibit problems. Because poorly executed or indicated maintenance works can have a significant impact on road safety and mobility, especially in cities, this will not only reduce the cost for the utility company, but also contribute to a healthier and safer environment for local residents.**

***Index Terms*—Subcontractor performance, Monte Carlo simulation, Bayesian Network, predictive maintenance**

I. Introduction

The work in this paper revolves around the following situation. Given a number of open maintenance sites managed by several subcontractors working for a same general contractor, an inspector from the contractor will be sent to these sites for inspection. This inspection is carried out by means of a tablet with a dedicated application that presents the inspector with a predefined set of questions, which are grouped per topic and mainly revolve around security and safety. The list of questions the inspector needs to answer for a specific inspection is decided upon beforehand, and revolves around chosen topics. In other words: either all questions belonging to a topic are presented, or none are. A specific question can be answered one of four ways: `ok` (the question is satisfied), `nok` (there was an issue), `sos` ("solved on site": there was an issue, but it was immediately resolved) or `na` (not applicable). The answers are communicated to a central server and stored for further processing or review.

The research question we answer in this paper is whether historical audits collected this way can be used to predict which currently open sites require more urgent inspection. We demonstrate that this is possible, and make available an end-to-end framework, for which the source code can be found at the project website[1].

This appears to be an understudied field. Some works suchs as [1] and [2] do deal with predicting subcontractor performance, but substantially differ from this work in both goal, approach and data used. To the best of our knowledge, this is the first work of its kind that proposes a simple method, using clearly defined data from a central and homogeneous source, with the specific aim to rank open sites according to expected performance.

II. Data

The data at our disposal are an anonymized historical list of audits, ranging in time from 2016-09-14 to 2018-09-25. The data was provided as a tabular file, where each row corresponds to an answered question together with an assortement of other fields such as the applicable subcontractor and site, and all rows are ordered chronologically. Mapping this data to individual audits needed to be done by relating this file to a separate file containing a list of audits, with the date they were performed.

The data comprises 9672 unique audits of 4569 unique sites from 21 different subcontractors. Each site is located in one of nine distinct areas, and relates to one of four distinct tasks. Fig. 1 depicts the number of sites per subcontractor, as well as the number of areas and tasks each were active in. Over all audits, answers are given to 152 unique questions, which are divided across 21 topics. Typically, questions are answered on a per-topic basis. In the remainder, we consider a question "answered" if it was answered anything but `na`. Fig. 2 shows how many times each question was answered, while Fig. 3 shows the ratio of answers per question. Questions are weighted, with weights decided upon by the general contractor. Specifically, for our data the weights were $\in (1, 5, 20)$.

A site hence relates three variables: a subcontractor, an area and a task. In the remainder of this text, we will refer to such a triplet as a *configuration*. An audit is then a combination of a site, a configuration and a (pre-)prepared list of questions that

This work was funded by the Flanders Innovation & Entrepreneurship TETRA project "Intelligent Analysis of Time Series".

[1] https://iiw.kuleuven.be/onderzoek/eavise/iats/home

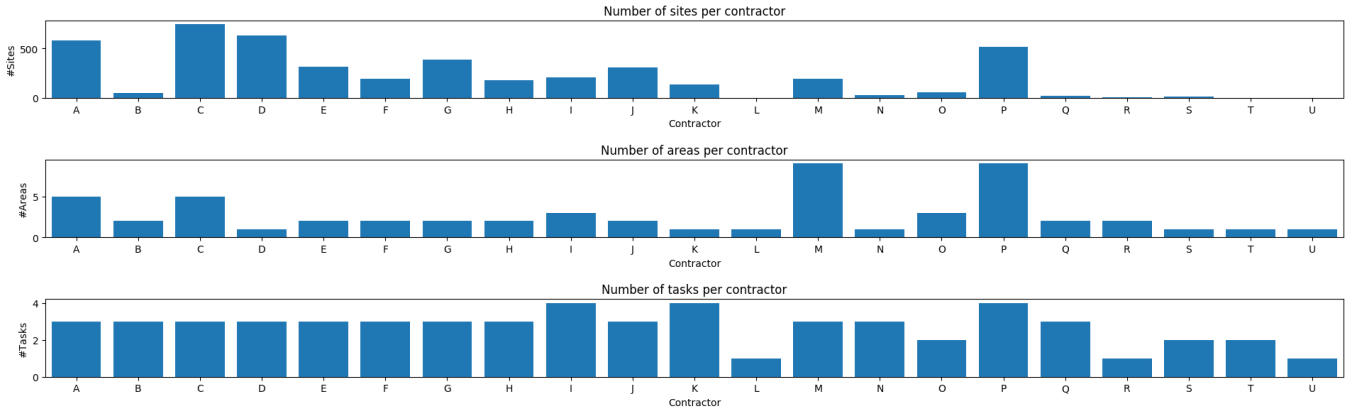

Fig. 1. Statistics per subcontractor.

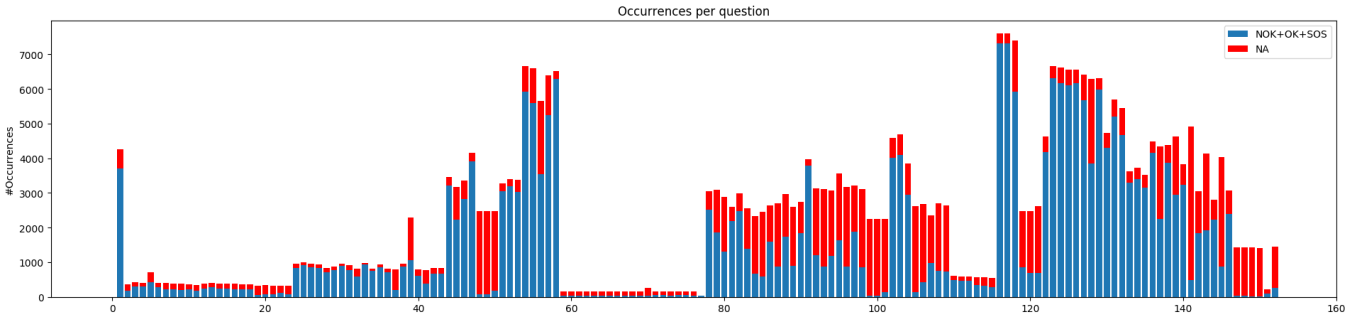

Fig. 2. How many times was each question answered?

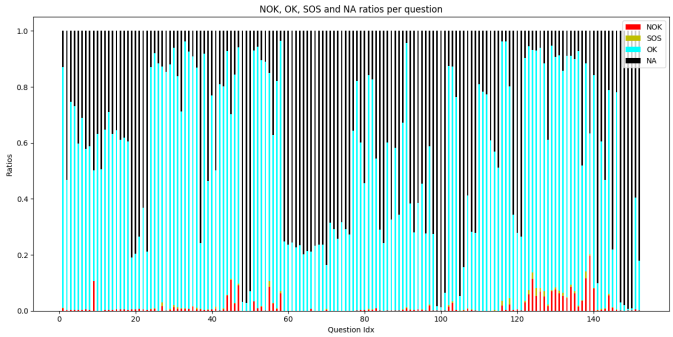

Fig. 3. Ratio of different answers per question.

relate to a number of chosen topics. For each audit, a score can be computed as follows, which we call the *deficiency*:

$$\text{deficiency} = \begin{cases} \dfrac{\sum_{i \in \{\texttt{nok}\}} w_i}{\sum_{i \in \{\texttt{nok},\texttt{ok},\texttt{sos}\}} w_i}, & \text{denom} > 0. \\ 1, & \text{denom} = 0. \end{cases} \tag{1}$$

where $\{\texttt{nok}\}$ denotes the set of indices of questions that were answered `nok` for that audit, and analogously for $\{\texttt{nok}, \texttt{ok}, \texttt{sos}\}$, and w_i is the weight of the question with index i. Put into words, the deficiency essentially answers the question: of all the questions that were answered, what ratio were answered `nok`?

III. Model Architecture

We model the auditing process as a random process. In other words, we assume the result of an audit, i.e., its deficiency, is drawn from a well-defined probability distribution. Our goal is to find a way to rank currently open sites according to our expectation of their deficiency. To do this, we will simulate the necessary probability distributions, and by comparing these, obtain the sought after ranking.

A. Critical Questions

Before moving on to the modeling proper, we need to introduce a new concept. As analysing of the data revealed that when the deficiency of an audit equals 1, typically only one or two questions were answered. Upon closer inspection these appeared to always be the same questions, which gave rise to the notion of a *critical question*: a question that when answered `nok` prompts the inspector to stop inspecting further. Let us now introduce two definitions. Note that looking at the data at site-level means we take all audits for that site together as if it were one single audit (with some questions possibly answered multiple times).

Definition III.1. A critical site/audit is a site/audit for which the deficiency = 1.

Definition III.2. A critical question for a specific critical site/audit is a question that was answered for this particular

critical site/audit (and hence, was answered nok). In other words, a question is only critical within the specific context of a critical site/audit.

Table I shows which questions were most often critical, i.e., answered for a critical site/audit. The results shown are limited to the 10 most answered questions, and do indeed suggest that some questions are more critical than others. Moreover, the most critical questions are identical whether we analyze at the site or audit level.

TABLE I
TOP 10 MOST COMMON CRITICAL QUESTIONS AT SITE AND AUDIT LEVEL.

Rank	Site		Audit	
	Question ID	Count	Question ID	Count
1	58	73	58	132
2	47	70	47	92
3	130	32	130	53
4	124	14	124	31
5	131	13	122	22
6	132	13	123	22
7	133	11	131	22
8	134	11	132	21
9	122	10	125	20
10	135	10	133	19
	Critical sites	Total sites	Critical audits	Total audits
	189	4569	325	9672

When narrowing down the analysis to the five most active contractors, the top 3 results per subcontractor are shown in Table II. Except for Subcontractor D, who has no critical sites, for all other contractors it is indeed apparent that there is always one question that is clearly more critical than others, namely question 47 for Subcontractor G, and question 58 for all others, with sites and audits agreeing. These are indeed also the two most critical questions as per the entire dataset.

TABLE II
TOP 3 CRITICAL QUESTIONS PER SUBCONTRACTOR. IF LESS ARE SHOWN, THERE WERE LESS THAN 3 CRITICAL QUESTIONS FOR THIS SUBCONTRACTOR.

Subcontractor	Site		Audit	
	ID	Count	ID	Count
C	58	43	58	61
	47	5	130	8
	130	4	47	132
D	–	–	130	1
A	58	5	58	5
	130	4	130	4
P	58	8	58	14
	122	2	122	5
	47	1	130	4
G	47	49	47	54
	124	3	124	5
	125	3	138	4

The next question is whether these questions are sometimes answered nok without this immediately resulting in a critical site or audit. The answer is: yes, quite often actually. For the top 5 most critical questions as per the site analysis (see Table I), we get the results shown in Table III. As these results show, most often a nok for these questions will not result in a critical audit, but nevertheless the top 2 questions are critical in more than 24% of the cases. Rephrased: when one of these questions is nok, there is almost a 25% chance of the audit being critical.

TABLE III
TOP 5 MOST COMMON CRITICAL QUESTIONS: HOW OFTEN ARE THEY CRITICAL?

ID	Critical sites	#Sites nok	Critical sites%	Critical audits	#Audits nok	Critical audits%
58	72	348	20.7	132	412	32.0
47	67	281	23.8	92	383	24.0
130	31	273	11.4	53	361	14.7
124	10	577	1.7	31	742	4.2
131	12	301	4.0	22	353	6.2

B. Modeling the Auditing Process

An audit is essentially a list of questions to be answered. As such, it can be modeled by randomly generating an answer to every question. The priors for each possible answer to each question can be extracted from the data, and are dependent on the subcontractor, area and task. A Bayesian Network lends itself perfectly to this type of application.

The preceding subsections suggest the following structure for our model. First of all, we decided to create two different networks: one for modeling performance dependence on area, the other for modeling performance dependence on task. An alternative would have been to use one network to model performance based on area and task together. However, we did not have enough data to reliably learn the parameters of the CPTs of such a network. Moreover, this problem is inherent to the problem at hand which puts a "physical" limit on the amount of data collectable per subcontractor – a subcontractor cannot serve 100 sites per day.

Our model is graphically depicted in Fig. 4. We start off with a root node representing the subcontractor, since this is the first variable that determines a configuration.

Fig. 4. Schematic of our Bayesian Network. 'CQx Ans' = 'answer to Critical Question x', 'CQx C?' = 'is Critical Question x critical?', 'RQ' = 'Remaining Question'.

Next up is either the area or the task, depending on which model we consider. This node is the only differing node between the two networks.

Once we have a subcontractor and an area or task, we can essentially begin filling in our questionnaire. But as we have seen in §III-A, sometimes the audit consists of only one or two critical questions. Hence, we chose to include this behavior in our model by means of a succession of three layers of nodes:

- The first layer contains as many nodes as there are critical questions considered in the model. This is a model parameter, and in this work we chose to use the top 3 critical questions. Each node represents the answer (`nok`, `ok`, `sos` or `na`) to a specific critical question.
- The second layer also contains one node per critical question, and represents whether the question is indeed critical or not. Note that it can only be critical if the answer to the question represented by its parent node in the first layer is `NOK`.
- The third and final layer contains a "Sum" node that simply encodes whether at least one of the critical questions is critical, i.e., whether at least one of its parent nodes has fired. If this is the case, then no further questions will be answered.

At this stage in the network, we know whether the site is critical or not, and hence, whether we need to answer the remaining questions or not. If so, we start by chosing whether or not to look at the questions belonging to a specific topic. This is encoded in a layer containing one node per topic which represents the odds of (the questions belonging to) this topic begin answered. If a topic node fires, all its children, which represent the questions belonging to that topic, will be answered (possibly `na`).

The final layer contains one node per question. Each node is connected to its associated topic node, and represents the odds of each possible answer for a question. The only exception is for critical nodes, for which an RQ-node is created that duplicates its state (see Fig. 4). Having these duplicate nodes makes it easier afterwards to process the audit code-wise.

Initializing a network amounts to determining the CPT for each node. This in turn is done by parsing the data on a per-audit basis, and keeping track of all node states. The CPTs can then trivially be derived from the counts of all states.

Once the network has been initialized, we can sample from the probability distribution it represents by randomly assigning a value to each node by sampling its corresponding CPT and respecting the hierarchy of the network. For each sample we can then compute the corresponding deficiency by looking at the final layer, i.e., the answers to all questions, and applying (1).

C. Scoring an Open Site

So far, we have two classes of models. One class represents subcontractor performance given a specific area, the other represents subcontractor performance given a specific task. Ultimately, given a specific configuration (subcontractor, area, task) by which we parameterize an open site, we want to use both these models to obtain one single estimate of the expected deficiencies, so that by comparing these estimates the corresponding sites can easily and straightforwardly be ranked.

The way we go about doing this is as follows. Given a certain configuration (subcontractor, area, task), we decompose it into two subconfigurations: (subcontractor, area) and (subcontractor, task). These two subconfigurations are then used to initialize the two uppermost layers of our models, after which the models are sampled. The number of samples to be drawn from each model is a parameter.

For each sample we draw from a model, we can compute the corresponding deficiency. To transform these sampled deficiencies into a distribution, we use a normalized histogram with a binwidth of 0.01. Hence, bins are the half-open intervals $[0, 0.01)$, $[0, 01, 0.02)$,..., except for the last interval which is the closed interval $[0.99, 1]$. This is essentially a Monte Carlo simulation.

To check the effect of the number of samples drawn on the stability of the generated distribution, we conducted the following experiment. For a specific model[2], for several values of the number of samples to be drawn, we generated 25 distributions. We then plotted the ratio of the standard deviation to the average over these 25 distributions as a function of the number of samples drawn, and obtained Fig. 5. As expected, the more samples are drawn to generate the distribution, the less the different simulations differ from each other. Time considerations made us decide to settle for 10,000 samples for our work. Our 8th Gen i7 CPU manages to generate about 100 random audits per second, meaning generating one distribution takes about 100s. Note that generating distributions can be parallelized.

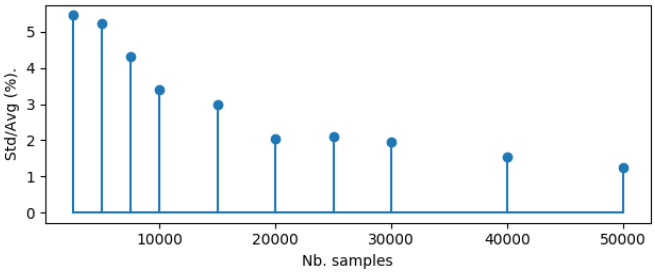

Fig. 5. Effect of the number of samples on robustness of generated distribution.

To compare distributions, we use what we call the *mass* of the distribution:

$$\text{mass} = \sqrt{\sum_{n=1}^{100} (0.01 \cdot n \cdot b_n)^2}, \tag{2}$$

where b_n represents the weight of the n^{th} bin. This metric thus ranges between 0.01 and 1, with the lowest end attained

[2]We chose the area model for Subcontractor C, although the exact model does not really matter in this case.

when all mass is contained in the first bin, and the highest end attained when all mass is contained in the last bin. The higher the mass, the more the distribution is concentrated towards 1, and therefore the higher the odds of obtaining a high deficiency when randomly sampling the distribution.

We now derive our estimate of the expected defificiency of the given configuration from two numbers: the mass of the histogram for the area subconfiguration, and that of the histogram of the task subconfiguration. In a sense, we consider these numbers to represent a probability, namely the probability of obtaining a high deficiency when randomly sampling the distribution. Combining both into one number to characterize the full configuration is then done, inspired by the statistical rule $P(A, B) = P(A) \cdot P(B)$, multiplying both numbers.[3] This ultimately gives us one number by which we express the belief that a certain configuration (Subcontractor, Area, Task) will result in a high deficiency. Given a set of such configurations, this number can then be determined for each configuration and used to rank them.

D. Lookback

We explored the idea of giving less weight to older audits. The rationale behind this is that the performance of a subcontractor varies over time and hence that more recent audits more accurately reflect the level of performance that can be expected of this particular subcontractor right now.

This concept, which we refer to as *lookback*, is graphically illustrated in Fig. 6. The figure shows a *cutoff date*, the date up until which the training data is used. The number of lookback days defines a timespan, starting from a given *cutoff date* and going back into the past, in which samples are unweighted. Going back further past this timespan, the weight of the samples lineary decreases. This results in the following rule:

$$\text{weight} = \begin{cases} 1 & \text{days past} \leq \text{lookback days} \\ \dfrac{\text{lookback days}}{\text{days past}} & \text{days past} > \text{lookback days} \end{cases} \tag{3}$$

with "days past" the number of days between the audit date and the cutoff date. Besides this, we also examined two variations that use either the square or the square root of this value. Space constraints prohibit an elaborate analysis, but the results in this paper use a square root weight.

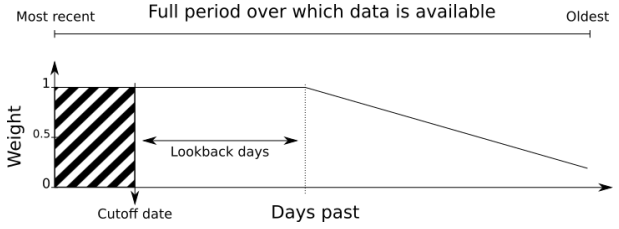

Fig. 6. Schematic illustration of lookback concept.

[3]Of course, this rule is only valid when A and B are independent, which isn't (necessarily) the case here, hence "inspired by".

E. Weighted Laplace Smoothing

When only little data is available for a given subcontractor, it is to be expected that many questions will never have been answered. In our approach, this would result in sparse CPTs, which in turn would mean that when sampling the model these unseen questions can never be answered, resulting in a distorted model. We solve this issue using the well known Laplace smoothing, concretely meaning we initialize all CPTs with a pseudocount instead of 0, be it with a twist. The relative weight of the pseudocount gains in relative importance as we add lookback. Consider, e.g., the case where we start giving less weight to audits that are older than 10 days past. In this case a pseudocount of 1 is already very high, compared to the case where all audits have a weight of 1, not just those of the past 10 days (i.e., the case without lookback).

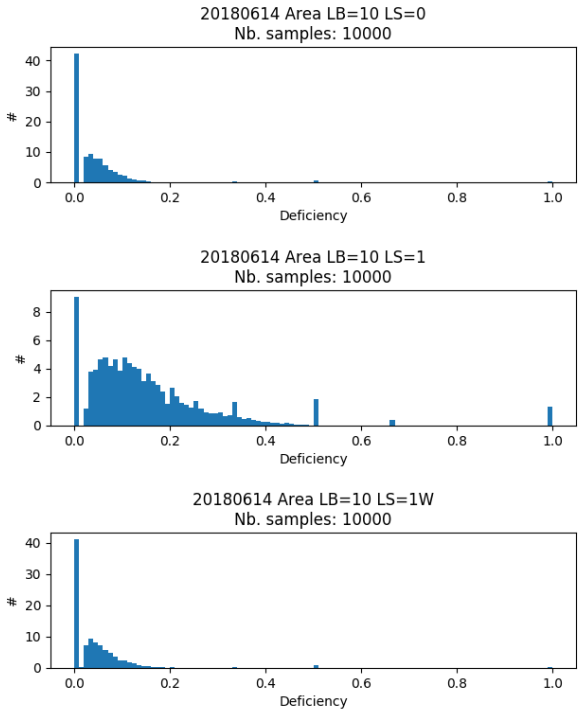

Fig. 7. Comparison of sampled distributions for Subcontractor A using a Lookback of 10 days. "LS=1W" refers to the weighted variant of Laplace smoothing.

For this reason we decided to implement "weighted Laplace smoothing". Basically, the idea is that we take as pseudocount the lookback weight of an observation on the oldest date in the train dataset. This way, the pseudocount gets weighted using the weighting mechanism described in §III-D, as if it was an audit that took place on that specific date.

Fig. 7 illustrates the effect of this weighted Laplace smoothing. The top graph shows a simulated distribution with 10 lookback days and no Laplace smoothing. The middle graph show the same, but with Laplace smoothing added with pseudocount = 1. As can clearly be seen, this heavily distorts the distribution. This problem is solved in the bottom graph,

TABLE IV

DEFICIENCY AT RANK k FOR TOP 5 CONTRACTORS FOR DIFFERENT TIME PERIODS. 'REF.' INDICATES REFERENCE ORDERING, 'PRED.' INDICATES PREDICTED ORDERING, '-1' INDICATES THERE WERE NOT ENOUGH AUDITS TO REACH THE DESIRED RANK. BEST RESULTS INDICATED IN BOLD.

Rank	2018-06-03		2018-06-17		2018-07-01		2018-07-15		2018-07-29	
	ref.	pred.	ref.	pred.	ref.	pred.	ref.	pred.	ref.	pred.
1	**0.048**	0.0	0.159	**0.500**	0.429	**0.800**	0.0	**0.147**	**0.161**	0.077
2	0.055	**0.500**	0.091	**0.330**	**0.614**	0.415	0.035	**0.108**	**0.125**	0.116
3	0.370	**0.373**	0.227	**0.273**	**0.420**	0.413	0.072	0.072	**0.134**	0.081
4	0.292	**0.381**	0.177	**0.330**	**0.317**	0.316	0.054	0.054	**0.120**	0.101
5	0.257	**0.316**	0.168	**0.269**	0.253	**0.339**	0.043	0.043	0.096	**0.097**
6	0.214	**0.274**	0.167	**0.274**	0.217	**0.449**	0.038	0.038	0.085	**0.095**
7	0.186	**0.244**	0.214	**0.254**	0.329	**0.403**	**0.033**	0.032	**0.087**	0.086
8	0.165	**0.220**	0.192	**0.347**	0.289	**0.357**	0.029	0.029	0.076	**0.087**
9	0.153	**0.201**	0.178	**0.311**	0.257	**0.317**	-1.0	-1.0	0.067	**0.078**
10	0.178	**0.183**	0.164	**0.286**	0.231	**0.290**	-1.0	-1.0	0.069	**0.070**

which uses a weighted Laplace smoothing, thus weighting the original pseudocount of 1.

Note that adding Laplace smoothing to our model in practice requires a bit of carefulness. Indeed, several entries in the CPTs correspond to configurations that should never occur in reality, and hence, should always be kept to 0. E.g., when the "Sum" node is "True", meaning we are dealing with a critical site, all Topic nodes should be "False", always. This required coding some custom rules in these specific cases.

IV. EVALUTION

To check the efficacy of our method, we wish to compare the rankings of sites as determined by our model for, say, a given day to the true ranking of the sites on that day according to their deficiencies. For this, we used an "average deficiency at rank k" approach as follows:

1) Train model with data up to date d.
2) Take all sites that were audited in the x days following d, with x a user-defined parameter.
3) Rank the sites according to our system, and according to the original order they were visited.
4) At rank k, compute the average deficiency of all sites of rank $\leq k$.
5) Compare the results for both orderings: the higher, the better.

Note that if more than one audit is present for a same configuration in the evaluation period, we take the average of the deficiencies of these audits to determine the rank of this configuration. All models used for evaluation use lookback days = 25 and weighted Laplace smoothing.

A. Main Results

Table IV summarizes our main results. It shows results of the same experiment repeated five times for five different dates. The date at the top of a column indicates the cutoff date used when training the model (always a Sunday), making the test dates the 10 working days following this date. Data was limited to the top 5 most active contractors to avoid data scarcity problems (see Fig. 1). These results show that our model often strongly outperforms the original order, meaning concretely that if our order were followed, many sites with poor audit results would have been visited earlier.

TABLE V

AVERAGE DIFFERENCE IN DEFICIENCY AT RANK k BETWEEN PREDICTED AND REFERENCE ORDERINGS.

1	2	3	4	5	6	7	8	9	10
.145	.110	-.002	.044	.049	.082	.034	.058	.050	.037

These results are summarized in Table V, which lists the average difference over the five periods between the deficiency at rank k for the predicted and reference orderings at every rank. The benefits of our system are clear, with a more than 10% increase in average deficiency for the top two ranks, and a 5% increase still apparent at lower levels.Note that as k grows larger, the top-ks according to different rankings will have more sites in common (in the extreme case, k equals the total number of sites and every ranking's top-k will consist of all sites). Therefore, it is to be expected that the difference between two rankings decreases as k grows.

V. CONCLUSION

This paper proposes an end-to-end framework that takes historic audits as input, and outputs a ranking of open sites ordered by their expected deficiency. To this end, the auditing process is modeled using a Bayesian Network whose node CPTs are initialized by extracting the corresponding statistics from the historic data. A Monte Carlo simulation is then performed using this network to sample the underlying probability distribution over the deficiency outcomes. We then rank these distributions by their mass. We demonstrate orderings by our system attain an increase in average deficiency of 10% at the top two ranks, with a 5% increase still apparent at lower levels.

ACKNOWLEDGMENT

We are grateful to TenForce[4] for providing us with the necessary data to carry out this work.

REFERENCES

[1] K. Chien-Ho, "Predicting Subcontractor Performance Using Web-Based Evolutionary Fuzzy Neural Networks," The Scientific World Journal, vol. 2013, pp. 9, 2013.

[2] Eddie W.L. Cheng, "Risk assessment in prospective performance prediction for subcontractors", Architectural Science Review, 59:2, pp. 126-135, 2016.

[4]https://www.tenforce.com/