RDI: A Real-time Decision Support System Applied to Dispatch Decision Problem

Raul S. Ferreira[1], Mauricio P. Dal Pont[1], Bruno M. A. da Silva[1], Wendell W. Teixeira[2]

[1]Radix Software & Engineering
Rio de Janeiro, RJ, Brasil

[2]CPFL Energy
Campinas, SP, Brasil

raul.ferreira,mauricio.dalpont,bruno.silva{@radixeng.com.br}

wendell@cpfl.com.br

Abstract. *Several important services with limited resources require uninterrupted support for a vast amount of customers. For example, internet providers, hospitals, distribution utilities and so on. When a customer's call is received in the proper channels, the reported problem pass through a screening phase in which the operator judge whether or not a support should be sent to the customer. However, not all problems are responsibility of the service provider. For instance, when a distribution utility sends a maintenance team to solve an energy issue that is out of company scope's, this action generates an improper dispatch problem. Improper dispatches bring high costs regarding fuel and logistics, and can result in heavy penalties to the company. For tackling this problem, we propose RDI, a decision support system that combines supervised machine learning algorithm and model predictive control (MPC) techniques. RDI receives customer's calls information and recommends when a maintenance team should be dispatched or not. Our first results indicate an assertiveness of 83% in the number of true positives (proper dispatches) and a decrease of 51% in the number of false positives (improper dispatches) within a real dataset from the industry. Moreover, RDI is capable of calculating the associated risk of each occurrence and by predicting changes in the current number of unsolved customer's calls using a Markov chain model. We show how we built this system, how this solution was applied for diminishing dispatch costs inside a distribution utility and possible directions for further research.*

1. Introduction

A distribution utility has to deal with several customer's calls regarding grid maintenance or energy issues. For instance, the customer's house energy suffers an interruption due to a failure in the distribution system. The agents receive these calls on a daily basis through proper channels. Generally, a customer's call is received by the agent and then this occurrence pass through a screening phase. Hence, the agents responsible for collecting data about the occurrence can analyze the customer information. Finally, the agents decide whether a maintenance team must be sent to the customer address to solve the problem.

However, not all problems are responsibility of the company. Thus, there are problems reported by the customers that are out of the company's scope. For example,

the customer's house energy turned off due to a fallen tree. Since the problem of fallen trees in a street is solved by a municipality agency, this problem does not belong to the company. This type of problem is denominated improper dispatch. The problem generates an unnecessary displacement for the maintenance team generate high costs regarding logistics and fuel since the problem contributes to wrong schedule planning and more travelled distance by the maintenance team. Moreover, a high number of improper dispatches can result in heavy penalties to the company since the maintenance team will not be available to customers that really need support, creating a delay in the maintenance services. The higher is the attendance delay, the higher is the penalty. Therefore, the decision to dispatch a maintenance team follows two questions: 1)What is the chance of the call be an improper dispatch ? 2)What is the cost if a maintenance team is not sent ?

To help the decision makers, we propose RDI, a real-time information system that uses supervised machine learning along with a model predictive control (MPC) technique. The supervised machine learning algorithm calculates the probability of the call be an improper dispatch while the MPC calculates the risk associated with the possible cost if a maintenance team is not sent. RDI uses the customer's calls information and historical data to recommend to the operation manager whether or not a maintenance team should go dispatched. The first version of the system was capable of achieving 83% of assertiveness regarding true positives (proper dispatches) and halved the number of false positives (improper dispatches). Thus, RDI is capable of drastically reducing operation costs.

Therefore, this work is organized as follows: in Section 2 we present related work regarding solutions for call center maintenance operations. In Section 3, we show the architecture of this system and its main modules. In Section 4 we present how we validated the RDI system through a real scenario in the energy industry. In Section 5, we discuss about the main points of the solution, and directions for further research.

2. Related Work

Regarding automatic classification for call centers, one of the first approaches was proposed by [Busemann et al. 2000] in which the authors apply text processing techniques along with machine learning algorithms within an assistance system for call center agents. Using learning algorithms such as support vector machines (SVM) [Hearst et al. 1998] the authors were able of achieving 78% of accuracy. Another study about classification applied to call center domain was proposed by [Tang et al. 2003] where the authors developed four methods for call-type classification using SVM on a database of human-to-human conversations recorded from an information technology help desk call center.

Several applications using machine learning algorithms were proposed for dispatch management. For instance, a support decision system for dispatching of trucks loaded with ready mixed concrete [Maghrebi et al. 2013]. The authors applied decision trees and rule induction and compared with an operation manager obtaining superior results. Another related work analyzes sound data collected by microphones attached to the businesses in edges [Yamato et al. 2017]. Thus, when an anomaly data is found, it automatically sends a maintenance team to the location. Another example of support decision system proposes to tackle predictive maintenance problems in the electric power distribution systems [Barriquello et al. 2017]. The authors intended to build a system capable of performing analysis, simulation, planning, and operation of maintenance and

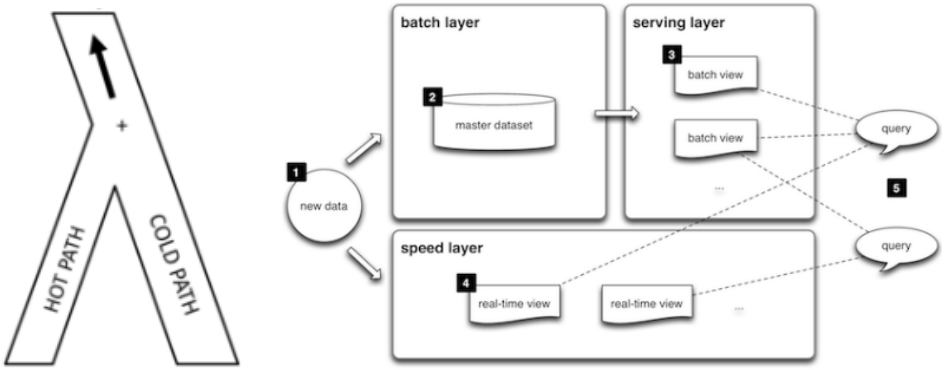
customer services in electric power distribution systems.

Another relevant related work is the decision support system built for helping to make decisions regarding problem of dispatching ambulances for emergency medical services [Roy 2017]. The authors propose a system that applies agent-based modeling and simulation concepts in evaluating different approaches to solve ambulance-dispatching decision problems under bounded rationality. Thus, it investigates the effect of over-responding, that is, dispatching ambulances even for doubtful high-risk patients, on the performance of equity constrained emergency medical services. The authors developed two different dispatching policies: first, a policy based on maximum reward, and second, a policy based on the Markov decision process formulation. The second policy was better since it solved the problem using value iteration method, performed better than the maximum reward method in terms of number of served patients.

3. The RDI Decision Support System

The RDI system was built to support operations centre in dispatch decision problems. Since the system uses historical data for training the supervised learning algorithm and receives real-time data regarding the incoming calls, the lambda architecture [Hausenblas and Bijnens 2018] was chosen for building RDI. Lambda architecture takes into account that a system has three layers: the hot path (real-time processing), cold path (batch processing), and the service layer, as illustrated in Figure 1(a).

Hence, the following flow can be applied in a lambda architecture: data is acquired (1) and distributed to the real-time processing layers (4) and for the batching processing; the real-time layer is a fast processing layer that ingest small amount of real-time data. On the other hand, the batch layer has a higher latency since it process much more data, generally, historical data. Later, this information is consumed in the service layer (3) through specific queries (5). Figure 1(b) illustrates the entire process.



(a) Lambda concept. (b) Lambda architecture.

Figure 1. Lambda model.

Since RDI was built to support operators that need to make decisions about maintenance dispatches 24 hours per day the system was built on top of the lambda architecture and the application flow is illustrated in Figure 2. In the first layer, RDI is hosted in the same workstation of the systems used by operators. The second layer contains an API responsible for communicating with the first layer and the third one. The API receives

actions from the RDI system and process the machine learning and MPC algorithms with data retrieved from the several databases from the operations centre. After that, the API returns the response to the RDI system.

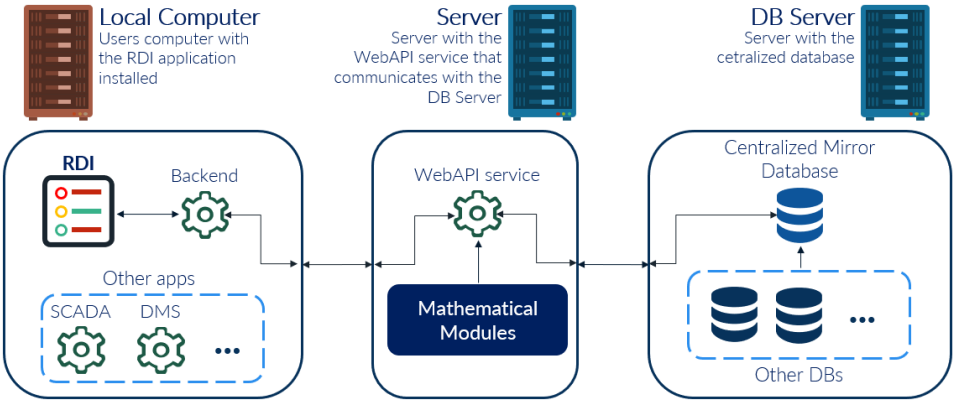


Figure 2. System communication.

The RDI system is composed by two modules: a machine learning module and a MPC module. The machine learning module is responsible for giving the probability of a proper/improper dispatch. The MPC module is responsible for calculating the risk of dispatch/not dispatch a support team. This risk is related to the cost associated if an operator does not send a support team to the problem. Furthermore, the risk is also associated with the importance of the call. For example, for a distribution utility, a hospital without energy is more important than a house with the same problem.

The machine learning module is composed by a supervised learning algorithm denominated extreme gradient boosting [Chen and Guestrin 2016]. The algorithm was chosen for its scalability since it runs up to ten times faster than another popular solutions on a single machine and scales to billions of examples in distributed or memory-limited settings. Besides, since this learning model is easily explainable, this algorithm seems to be suitable for applying in generic fields with convincing results. The probability of proper/improper dispatch calculated by the machine learning module is also used as part of the cost function in the MPC module.

The MPC module is responsible for evaluate the costs to send or not a support team to support customers. The MPC is an algorithm that tries to obtain a control law that minimizes an objective function [Camacho and Bordons 2007]. Building a MPC control strategy demands three main items: a predictive model, an objective function and a procedure to obtain the control law [Normey-Rico and Camacho 2007]. For this work, the predictive model was designed using a Markov chain process, which is a discrete model where the next state only depends on the current state. The Markov property states that the process is memoryless, that is, the system is a stochastic process [Bogachev 2006] defined as:

$$P(X_t = j | X_0 = i_0, X_1 = i_1, ..., X_{t-1} = i_{t-1}) = P(X_t = i_t | X_{t-1} = i_{t-1})$$

The objective function needs to represent the risks of dispatching or not an support team. Also, it needs to represent the total risk of the current state looking forward on the prediction horizon. Hence, it is possible to prioritize the events. The objective function

$J = \sum_{i=1}^{M} J_i + \sum_{1}^{N} \sum_{p=1}^{P(N)} J_p$ (1) is a sum of every event objective function on current state $J_i = \lambda_i u_i + \overline{\lambda_i} R(ID_i, H_i)$ (2) and in the future states $J_p = \lambda_p u_p + \overline{\lambda_p} R(H_p)$ (3). In Equations 2 and 3, u is the cost of dispatching a maintenance team to solve the problem and R is the cost function of not dispatching a support team to that event with arguments ID. Therefore, ID is the machine learning output probability of the event being an improper dispatch and H the total time that the event is unsolved. Besides, λ is the binary control operator that decides to dispatch (1) or not dispatch (0) a support team. In Equation 1, the second part refers to the costs on the predicting horizon N, where the only input is the total time that the event remains unsolved. The variable P stands for the number of events predicted on that state N. Minimizing Equation 1 gives us the control array $\boldsymbol{\lambda}$ that decides when to dispatch or not an event considering constraint of finite resources (number of support teams).

4. Experimental Settings

We validated the RDI application in a real scenario: the operations centre of the distribution utility CPFL Energy. The dataset chosen for this work was extracted from the call center databases from CPFL Energy. The extracted dataset contains 248,570 registers with one year of customer's calls. Each register has five features: neighbourhood, city, property type, number of recent calls and event priority. The dataset are labeled with two classes: improper dispatches and proper ones. The classes are imbalanced: 80% of proper dispatches and 20% of improper dispatches. In the next subsections we explain the experiments and the results for the machine learning and MPC modules.

4.1. Experiments

For machine learning experiments, data was divided in 80% for training and 20% for testing. Beyond the overall accuracy, the chosen metrics to assess the results were macro-F1, for measuring the quality of intraclass classification, and the Matthews correlation coefficient (MCC) [Matthews 1975] to measure how the system classifies the minority class, since this dataset is imbalanced. For the Markov chain model, data was divided in 20% for training and 80% for tests, and the predicting horizon of the MPC's objective function was set to a short sighted $N = 1$ for this first experiment. That means the Markov chain will predict the changes on the current number of events just on the next state.

4.2. Results

Table 1 shows the confusion matrix of the results of the machine learning model. The quantity of improper dispatches (false positives) was halved while the dispatches that really need support (true positives) suffered a small decrease, about 17%.

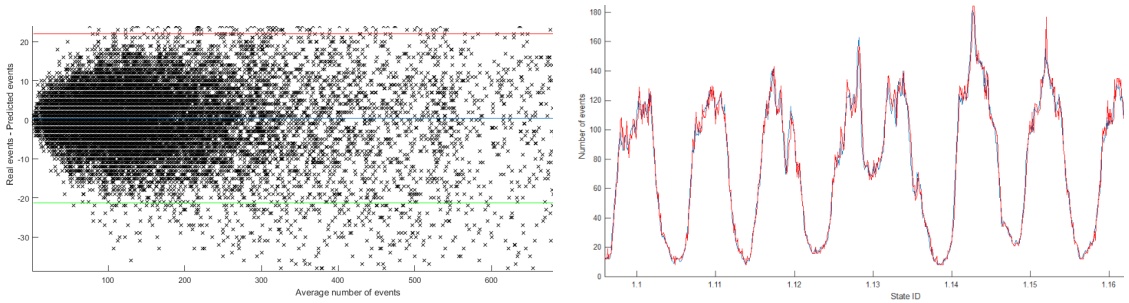
Table 1. Confusion matrix dispatches

Dispatches	True positive	False positive	True negative	False negative
248,570	57,440 (83%)	11,586 (17%)	2,828 (51%)	2,717 (49%)

Even the overall accuracy was 80.82%, the MCC result was 0.23. These results corroborate that the system correctly classifies a high number of true positives but still has difficulty to correctly classify true negatives. However, the macro F1-score was 0.59,

what indicates a reasonable quality in the predictions since the model achieves good precision values sooner, in the first events and decreases while the events come through the time. This behaviour indicates that data has concept-drift characteristics and possible approaches to deal with this are mentioned in Section 5.

Regarding the Markov chain model results, the process resulted in a high correlation $R = 0.9982$ with a percentage mean error of 5.9085%. However, a good correlation not always means a good agreement between datasets. Thus, the Bland-Altman plot [Altman D G 1983] was performed and its results are shown in Figure 3(a). This plot shows the agreement between real data and the predicted data. Using 1.96 as standard deviation, we can affirm that 95% of the predicted values will be within -21.2596 and 22.0874 of the real values. Hence, it is assumed that there is a consistent error of 0.4139 regarding predicted events. These analysis confirm that the Markov chain model is a reliable predictor on the $N = 1$ predicting horizon, as shown in Figure 3(b).



(a) Bland-Altman: limits between -21.2596 and 22.0874. Mean error = 0.4139.

(b) Comparison between the real dataset (blue) and the predicted dataset (red).

Figure 3. Markov chain model results.

Table 2 presents the preliminary results of the dispatch decision by the MPC. It shows the decision made at a certain state, where 59 events were currently unsolved with a limited number of 25 maintenance teams (constraint). The "unknown" on the last 5 events refers to the predicted events from the Markov Chain with $Np = 1$, which are not currently opened and the machine learning has not classified yet. It also predicts that all 5 events will not be improper dispatches. Using a Mixed-integer linear programming (MILP) [Bemporad and Morari 1999] algorithm, Equation 1 was minimized in order to obtain the binary control array $\boldsymbol{\lambda}$.

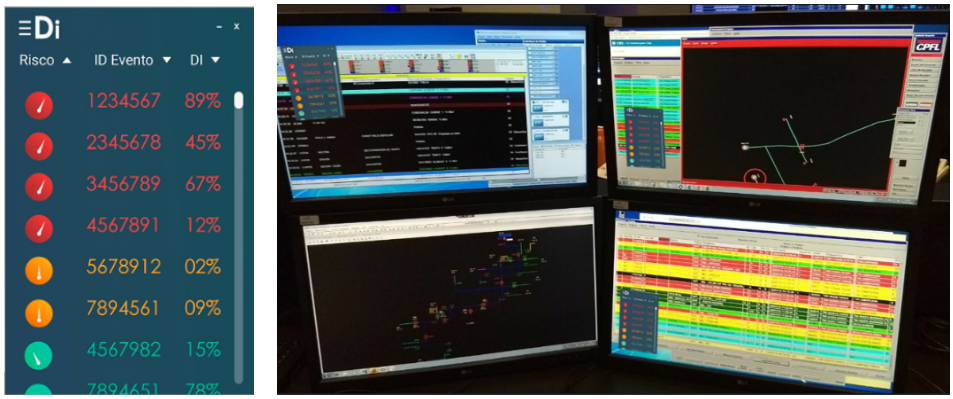
Each individual cost from each event on Table 2 represents the total costs of not dispatching a maintenance team. The minimization algorithm chooses the best setup resolving the highest risk events. MPC also decides to save 5 maintenance teams to solve the next 5 predicted events, which were higher risk events in that prediction horizon. Finally, Figure 4(a) shows the visual details regarding the application. The design of the application was made to not interfere with the existing systems the operations centre. Figure 4(b) shows the RDI application on the screens of the operations centre.

5. Conclusions

In this work we introduced a decision support system, denominated RDI, built for reducing the quantity of improper dispatches. We showed that combining techniques form two

Table 2. Decision for dispatching (1) or not dispatching (0) a maintenance team.

Event	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
% I.D.	45.05	76.55	56.88	60.28	50.46	52.27	41.87	16.37	58.36	47.85	66.88	45.97	35.18	73.77	74.57	16.92
Cost	1.58	1.13	0.60	1.26	0.95	0.77	1.14	1.06	0.61	0.82	0.49	1.27	1.00	0.75	0.49	1.78
Label	1	1	0	1	0	0	1	0	0	0	0	1	0	0	0	1
Event	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
% I.D.	40.03	38.54	47.80	71.15	45.04	68.38	73.18	66.32	70.24	54.04	35.71	63.62	43.31	26.39	60.83	38.82
Cost	1.52	1.07	1.36	0.96	0.96	1.30	0.46	0.57	0.96	1.39	1.01	0.69	1.19	0.88	0.65	1.65
Label	1	0	1	0	0	1	0	0	0	1	0	0	1	0	0	1
Event	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
% I.D.	53.55	70.72	28.08	55.47	83.00	71.28	59.58	41.53	72.07	45.65	39.42	21.36	68.92	69.86	38.33	26.38
Cost	0.64	0.62	1.63	1.27	0.28	0.45	0.65	1.61	0.72	0.87	1.44	1.01	0.76	0.63	0.87	1.60
Label	0	0	1	1	0	0	0	1	0	0	1	0	0	0	0	1
Event	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
% I.D.	71.49	38.51	71.85	12.88	76.53	61.80	71.14	11.83	46.23	59.77	25.47	unknown	unknown	unknown	unknown	unknown
Cost	1.22	1.09	0.56	1.07	0.77	0.55	0.46	1.15	0.69	0.67	1.29	1.25	1.25	1.25	1.25	1.25
Label	1	0	0	0	0	0	0	1	0	0	1	1	1	1	1	1



(a) RDI screen. (b) RDI on the existing systems screens.

Figure 4. RDI in production.

different fields such as machine learning and model predictive control bring interesting and reliable results for resource planning problems. We validated the RDI through a real scenario and the system helped not only to highly reduce the dispatch costs but improved the logistics of the operations centre.

Regarding machine learning module, RDI was able to reduce the number of improper dispatches up to 51% while keeping an overall accuracy of 83% for the proper dispatches. However, is expected that 100% of proper dispatches needs to be identified, thus improvements on the model are needed. Regarding MPC module, it predicted that 5 high risk events would be opened on the next state and decided to save maintenance teams to solve them later. However, expanding the prediction horizon to $Np > 1$ should present better dispatch strategies and needs more experimental tests. Finally, we highlight some directions for further research: 1)Density approaches aiming to reduce the bias of the learning model induced by the similar customer's calls contained in the stream [Ferreira et al. 2018a]; 2)Since some characteristics of this real-time dataset are non-stationary, learning models for concept-drift can improve the results [Ferreira et al. 2018b]; and 3)Development of better MPC objective function with other assumptions such as the total time that a customer does not have energy by month and the customer's location.

References

Altman D G, B. J. M. (1983). Measurement in medicine: the analysis of method comparison studies. *The Statisticians*, pages 307–317.

Barriquello, C. H., Garcia, V. J., Schmitz, M., Bernardon, D. P., and Fonini, J. S. (2017). A decision support system for planning and operation of maintenance and customer services in electric power distribution systems. In *System Reliability*. InTech.

Bemporad, A. and Morari, M. (1999). Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427.

Bogachev, L. V. (2006). Random walks in random environments. *Elsevier Encyclopedia of Mathematical Physics*.

Busemann, S., Schmeier, S., and Arens, R. G. (2000). Message classification in the call center. In *Proceedings of the sixth conference on Applied natural language processing*, pages 158–165. Association for Computational Linguistics.

Camacho, E. F. and Bordons, C. (2007). *Model Predictive Control*. Springer.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM.

Ferreira, R. S., Accioli, B., William, W., Zimbrão, G., and Alvim, L. (2018a). Density-based core support extraction for non-stationary environments with extreme verification latency. In *7th Brazilian Conference on Intelligent Systems (BRACIS)*.

Ferreira, R. S., Zimbrão, G., and Alvim, L. G. M. (2018b). *AMANDA: Density-based Adaptive Model for Non-stationary Data under Extreme Verification Latency Scenarios*. PhD thesis, Universidade Federal do Rio de Janeiro (UFRJ).

Hausenblas, M. and Bijnens, N. (2018). Lambda architecture - available from internet: http://lambda-architecture.net/ .

Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J., and Scholkopf, B. (1998). Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28.

Maghrebi, M., Sammut, C., and Waller, T. (2013). Reconstruction of an expert's decision making expertise in concrete dispatching by machine learning. *Journal of Civil Engineering and Architecture*, 7(12):1540.

Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451.

Normey-Rico, J. E. and Camacho, E. F. (2007). *Control of Dead-time Processes*. Springer.

Roy, R. B. (2017). Equity-constrained dispatching models for emergency medical services. *Team Performance Management: An International Journal*, 23(1/2):28–45.

Tang, M., Pellom, B., and Hacioglu, K. (2003). Call-type classification and unsupervised training for the call center domain. In *Automatic Speech Recognition and Understanding Workshop*.

Yamato, Y., Fukumoto, Y., and Kumazaki, H. (2017). Predictive maintenance platform with sound stream analysis in edges. *Journal of Information processing*, 25:317–320.