Predicting Call Center Performance with Machine Learning: Proceedings of the 2018 INFORMS International Conference on Service Science

Chapter · January 2019

DOI: 10.1007/978-3-030-04726-9_19

3 authors:

Siqiao Li
Vrije Universiteit Amsterdam
8 PUBLICATIONS **50** CITATIONS
SEE PROFILE

Qingchen Wang
The University of Hong Kong
4 PUBLICATIONS **26** CITATIONS
SEE PROFILE

Ger Koole
Vrije Universiteit Amsterdam
239 PUBLICATIONS **7,995** CITATIONS
SEE PROFILE

Predicting Call Center Performance with Machine Learning

Siqiao Li[1], Qingchen Wang[*1,2], and Ger Koole[1]…

Department of Mathematics, Vrije Universteit Amsterdam, Amsterdam, 1083HV, Netherlands

Amsterdam Business School, Plantage Muidergracht 12, Amsterdam, 1018 TV, Netherlands

Abstract

In this paper we present a simulation-based machine learning framework to evaluate the performance of call centers having heterogeneous sets of servers and multiple types of demand. We first develop a simulation model for a call center with multi-skill agents and multi-class customers to sample quality of service (QoS) outcomes as measured by service level (SL). We then train a machine learning algorithm on a small number of simulation samples to quickly produce a look-up table of QoS for all candidate schedules. The machine learning algorithm is agnostic to the simulation and only uses information from the staff schedules. This allows our method to generalize across different real-life conditions and scenarios. Through two numerical examples using real-life call center scenarios we show that our method works surprisingly well, with out-of-sample fit (R-squared) of over 0.95 when comparing the machine learning prediction of SL to that of the ground truth from the simulation.

1. Introduction

Call centers, as a typical large-scale service system, has attracted attention from the operations research (OR) community for more than 20 years. An important problem in OR is workforce scheduling, which in general can be categorized into 4 steps [1]:

1. **Forecasting:** predicting the arrival rates of each customer type over the planning horizon based on historical data.
2. **Staffing:** determining the minimum number of agents needed to meet a required quality of service (QoS) in each period (e.g., intervals). In practice, QoS can be measured in various ways such as the abandon rates, average response time (i.e., waiting time), etc.
3. **Shift scheduling:** determining the number of agents assigned to each shift for each agent group.
4. **Rostering:** assigning employees to the shifts.

In this paper we focus on steps 2 and 3 and introduce a simulation-based machine learning framework to predict QoS measurements. Our method can lead to an efficient way of solving the staffing and shift scheduling problem integrally.

A fundamental challenge in staffing and scheduling service systems is reaching certain QoS targets at minimum costs. This challenge becomes particularly complicated when considering call centers that have multi-skill agents and multi-class customers with heterogeneous arrival rates, as we need to decide the staffing level with the best configuration of multi-skill agents over the planning horizon with consideration of shift types. To exploit the flexibility to choose staffing levels in different time periods (e.g., intervals) so that the scheduling costs can be minimized, we consider the staffing and shift scheduling problem integrally.

Due to the complexity, few papers have studied the integrated problem in a multi-skill multi-class setting. The key difficulty is the lack of closed-form expressions for QoS measurements. To cope with this, the papers that study the integrated problem either rely on simulation [2-3] or approximate the QoS by modeling the system into certain existing mathematical models such as fluid models or queuing models [4]. The advantages of using simulation are multi-folded: it tends to produce more reliable results [4-5], takes into consideration the transient effects between intervals, allows for more diversified QoS measurements, and makes fewer assumptions so as to be more realistic. On the contrary, LP approximations which mostly based on fluid models and queueing models, often need strict conditions

* Qingchen Wang: Tel.: (+31) 636368068; E-mail: Q.Wang@uva.nl

such as heavy-traffic systems, short service times, impatient customers, and specific routing policies. Thus it can be less robust and is difficult to apply in practice. Moreover, modern call centers are not dealing only with calls, but also include other channels such as email and chat. Unfortunately, simulation methods can be very time consuming, which is a significant drawback in practice as it is common for practitioners to iterate multiple times between shift scheduling and rostering. In these cases, it is important to have a reliable staffing and shift scheduling method with short computation times.

As a result, we wonder whether there is an approach that can efficiently (i.e., quickly with acceptable accuracy) estimate the QoS measurements without strict assumptions. This paper proposes the following approach: introducing a machine learning algorithm that is trained on simulation results to predict the QoS measurements. More specifically, we first develop a general simulation model for call centers, and under any given scenario, we randomly generate a number of possible schedules with their corresponding QoS. The computational burden in this stage is not heavy as it does not need to be in real time. Subsequently, we train a machine learning algorithm on these schedules and are then able to quickly produce a "look-up table" with the QoS of all possible schedules. In this way, we can take advantage of the simulation method without having to bear the costs of long computation times.

2. Model

In this section, we provide a general description of our problem, based on which a simulation model is built.

We consider a call center where arriving customers are categorized in I types of service: $\{1, ..., I\}$. These customers can be calls, emails or chats that are served by agents divided into G groups: $\{1, ..., G\}$. Agents within the same group have the same skill set, which gives the subset of service types that this agent can serve. Finally, there are K different shifts: $\{1, ..., K\}$. The schedule then can be represented by $n_{g,k}$ which is the number of agents staffed to group g, and shift k. With respect to the QoS measurement, we choose the service level (SL), which is most commonly used in call centers. It is defined as the proportion of customers who wait less than a given time threshold, also known as the acceptable waiting time (AWT), over a time period. In practice, managers often pay attention to daily or weekly SL of each call type. Sometimes, they also evaluate several service types together as a set. In future work, other QoS measurements can be included easily if needed.

We do not assume any particular arrival process, service time distribution, or patience distribution, instead we only need to be able to simulate them. In practice, the arrival rates are normally derived from the forecast results, and redials and reconnects can also be included. We only assume that service within the same type will be served in a First Come First Served (FCFS) basis.

In the multi-skill multi-class environment, the choice of routing rules is also important for achieving good system performance. Again, we do not need to restrict our model to a specific routing policy. Later, we present the performance of our proposed approach by comparing two scenarios derived from real call centers, where static priority is used and preemption is allowed for some call types: once an agent starts serving a call, there can be an interruption by other calls with higher priorities.

The goal of this paper is to approximate the simulation with a machine learning algorithm so that a near optimal schedule can be found efficiently. To do so, once the simulation model is validated, we can model the simulation outcomes as a prediction problem, and then train a machine learning algorithm to predict SL outcomes for each scenario and schedule.

3. Machine Learning Approach

In order for the formulation to be generalizable to different scenarios and be invariant to simulation details, we only use information relating to the scheduling decision variables. Since SL is a continuous variable between 0 and 1, we model this as a regression problem and minimize the sum of squared error (SSE) between the true and predicted SL values of simulated samples during the training process. The general formulation of our problem can be written as:

$$arg_f min \sum_{s=1}^{S} (f(n_{1,1}^s, n_{1,2}^s, ..., n_{1,k}^s, n_{2,1}^s, n_{2,2}^s, ..., n_{2,k}^s, ..., n_{G,1}^s, n_{G,2}^s, ..., n_{G,K}^s) - SL_{true}^s)^2 \tag{1}$$

where S is the total number of simulated samples, and $n^s_{g,k}$ is the number of staff for skill group g scheduled in interval k.

Note that $n^s_{g,k}$ can be derived easily from the agent schedules.

f is a function that produces a SL prediction for inputs of a given staffing policy. An example of a function f that can minimize the SSE is least squares regression, where optimal weights for a linear combination of the staffing variables are computed. However, although we expect changes in SL to be monotonic with changes in the number of staff scheduled, least squares regression is not a suitable function to approximate the call center staffing problem due to non-linearity of SL in response to the number of staff.

We use Gradient Boosted Decision Trees (GBDT) for this problem [6]. It is also known by other names such as Multiple Additive Regression Trees (MART), Gradient Boosting Machine (GBM) or Tree Boosting. They all refer to the same technique which applies an algorithm called Gradient Boosting that uses classification or regression trees (CART) as base learners [7]. GBDT is an iterative algorithm and it works by training a new regression tree for every iteration to minimize the residual of predictions made by the previous iteration. The predictions of the new iteration are then the sum of the predictions made by the previous iteration and the prediction of the residual made by the newly trained regression tree in the new iteration.

GBDT is one of the most powerful machine learning algorithms and has been used to win most of the recent predictive analytics competitions [8]. It is also well suited for the problem of predicting SL from staff schedules. Unlike least squares regression, GBDT is both non-linear and non-parametric, and is able to exploit interactions between input variables. In this paper we use a fast and accurate implementation of GBDT called LightGBM, which is currently used in most solutions to data science challenges [9].

4. Numerical Experiments

We perform numerical experiments using two simulation scenarios to test the performance of machine learning predictions of SL in approximating the SL outcomes from the simulations. Both scenarios are derived from real-life call centers. The first scenario is fairly simple and is meant to be an easy case for the machine learning approximation. The second scenario is complex and is intended to test the capabilities of the machine learning solution in large-scale and highly complicated scenarios. In our case, we are trying to evaluate the SL of given schedules so that useful training data is expected to have well-distributed SL outputs and a good coverage of all shift types. To avoid simulation runs with poor coverage, we need to decide some scheduling parameters based on the given scenarios before generating random schedules: the upper/lower bound (U, L) of the total assigned agents, and the maximal/minimal (u, l) number of agents assigned to each combination of shifts and agent groups. Moreover, the agents are assigned to the combinations in a random order to avoid bias.

Scenario 1 is from a mid-sized English call center that has only inbound calls, in five different languages. Five corresponding agent groups are considered, one for each language, but the non-English groups are also able to handle English calls but with a lower priority. No interruptions are allowed. We randomly simulate 10000 schedules within the constraints of U=150, L=20, u=10, and l=0. Weekly SLs are measured by the proportion of the customers who wait less than their corresponding AWTs among all customers of a week.

Scenario 2 is designed by combining several mid-sized call centers. In total, we have 29 service types and 23 agent groups. Service types include 16 inbound call channels, 6 chat channels, and 7 email channels. The attributes of these service types vary greatly, including workload, arrival profiles, average service time, and patience. This results in highly complex relationships between staff schedules and SL. Chats, calls, and emails are differentiated as follows.

Chats and calls are real time, but multiple chats can be handled in parallel by a single agent. When the maximum number of parallel chats is limited (e.g., to 2 or 3), the service time distributions will not depend much on the current level of concurrency. The reason is that customers also need time to reply and this time can be used by the agent to respond to other customers. Emails are not answered in real time so customers do not abandon. Usually the AWT of emails is much longer than calls so they do not need to be handled within the interval of arrival. However, the waiting time of an email customer also includes the service time since they “wait” until they receive the answer.

Among the 23 agent groups, most of the groups' skill sets are overlapping and 4 groups are independent from others. The routing rule we used is still a static priority policy, but some emails can be interrupted by calls. Since this scenario is much bigger than the first one, we generated 20000 random schedules with U=500, L=20, u=100, l=0.

5. RESULTS

We present the results of the numerical experiments. For each scenario, we randomly selected 70% of the simulated samples to train the machine learning algorithm and test its performance on the other 30% of the samples. GBDT is able to perfectly fit any dataset it is trained on, so we must hold out a subset of the original dataset to test its performance. The goal of this paper is not to assess the ability of GBDT to reproduce simulation outcomes for samples it has already seen, but for samples it has not seen. The 30% of the samples that are held out can also be interpreted as how well GBDT can approximate the simulation in practice.

In order to improve the performance of GBDT we also included two additional sets of variables, $v_1^s = \sum_{k=1}^{K} n_k^s$ for each skill group g and $v_2^s = \sum_{g=1}^{G} n_g^s$ for each shift k to function f in equation (1). These variables are aggregates of the staffing numbers in equation (1) and helps to guide GBDT to better solutions with fewer training iterations.

Table 1: GBDT Performance on Simulation Data

Evaluation Metric	**Scenario 1**	**Scenario 2**
MAE	0.035	0.013
MAPE	12.46%	34.51%
WAPE	4.72%	11.67%
R^2	0.977	0.955%
N	3000	6000

Table 1 presents the performance of GBDT on predicting SL for the numerical experiment. For the 30% of the simulated samples that we use to evaluate performance, we compute the mean absolute error (MAE), mean average percent error (MAPE), weighted average percent error (WAPE), and the coefficient of determination R^2 between the predicted and actual SL outcomes.

The error rate in Scenario 1 is quite low with MAE of 0.035, while the MAPE is substantially higher at 12.46%. This is due to the samples with low SL which have higher MAPE values as compared to MAE. The 4.72% WAPE is more telling as it weighs samples by their SL values, and the predictions have an R^2 value of 0.977 which shows a strong fit. Overall this suggests that GBDT is able to perform very well on predicting SL for small-to-mid-sized call centers that are similar to Scenario 1. GBDT's performance on Scenario 2 is more complicated. Although MAE is low at only 0.013, the MAPE is very high at 34.51% due to large number of samples with very low SL, and even the WAPE is very high at 11.67%. However, the low SL is a result of how complex Scenario 2 is, and even though GBDT's performance is worse, it is still a better option than relying on simulations in real time to find acceptable solutions due to the large parameter space of 29 service types and 23 agent groups.

6. CONCLUSION

This paper presents a machine learning approach to evaluating call center staffing schedules for the integral staffing and shift scheduling problem. This is a challenging problem due to the lack of closed-form QoS measurements which leaves simulation as the best existing method to evaluate staffing schedules. However, simulation is slow and therefore cannot be used for real-time optimization of staffing schedules for large call centers, leaving faster methods to be desired. Our approach uses Gradient Boosted Decision Trees (GBDT) to train a prediction model on simulation samples that are generated offline for the purpose of evaluating staffing schedules online. We show that GBDT performs very well on a numerical example of a small-to-mid-sized call center and moderately well on a numerical example of a large and complex call center. Both are promising and can potentially allow for real-time optimization of staffing and shift schedules.

References

[1] Atlason J, Epelman MA, Henderson SG, "Optimizing call center staffing using simulation and analytic center cutting-plane methods", Management Science 54(2):295–309, 2008.

[2] Cezik MT, L'Ecuyer P, "Staffing multiskill call centers via linear programming and simulation", Management Science 54(2):310–323, 2008.

[3] Avramidis AN, Chan W, Gendreau M, LEcuyer P, Pisacane O, "Optimizing daily agent scheduling in a multiskill call center", European Journal of Operational Research 200(3):822 – 832, ISSN 0377-2217, 2010.

[4] Bodur M, Luedtke JR, "Mixed-integer rounding enhanced benders decomposition for multiclass servicesystem staffing and scheduling with arrival rate uncertainty", Management Science 63(7):2073–2091, 2017.

[5] Ingolfsson A, Campello F, Wu X, Cabral E, "Combining integer programming and the randomization method to schedule employees", European Journal of Operational Research 202(1):153 – 163, ISSN 0377- 2217, 2010

[6] Friedman JH, "Greedy function approximation: A gradient boosting machine", The Annals of Statistics 29(5):1189–1232, 2001.

[7] Breiman L, Friedman J, Stone CJ, Olshen RA, "Classification and regression trees" (CRC press), 1984.

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

[9] Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, Ye Q, Liu TY, "Lightgbm: A highly efficient gradient boosting decision tree", Advances in Neural Information Processing Systems, 3149–3157, 2017.