Les Cahiers du GERAD ISSN: 0711–2440

Waiting time predictors for multi-skill call centers

M. Thiongane, W. Chan,
P. L'Ecuyer

G–2015–112

October 2015

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Waiting time predictors for multi-skill call centers

Mamadou Thiongane
Wyean Chan
Pierre L'Ecuyer

GERAD & Department of Computer Science and Operations Research, Université de Montréal, Montréal (Québec) Canada, H3C 3J7

mamadou.thiongane@umontreal.ca
chanwyea@iro.umontreal.ca
lecuyer@iro.umontreal.ca

October 2015

Les Cahiers du GERAD
G–2015–112

Abstract: We develop customer delay predictors for multi-skill call centers that take as inputs the queueing state upon arrival and the waiting time of the last customer served. Many predictors have been proposed and studied for the single queue system, but barely any predictor currently exists for the multi-skill case. We introduce two new predictors that use cubic regression splines and artificial neural networks, respectively, and whose parameters are optimized (or learned) from observation data obtained by simulation. In numerical experiments, our proposed predictors are much more accurate than a popular heuristic that uses, as a predictor, the delay of the last customer of the same type that started service.

1 Introduction

1.1 Context and problem

We study delay predictors for multi-skill call centers, where the goal is to estimate the time that a customer, upon arrival, must wait before starting service with an agent. In multi-skill centers, customers are categorized by call types, and agents are divided into groups based on the subset of call types that they have the ability to handle (which defines their skill set). An agent can serve a customer only if he/she possesses the skill for that call type. Gans et al. (2003) give a thorough description of the operational aspects of call centers. In this paper, we use the words "customer" and "call" interchangeably, and the same for "server" and "agent".

A major contrast between the multi-skill and the traditional and well-studied single-skill system (one call type and one agent group) is the importance played by the routing policy on the dynamics of a multi-skill call center. The router manages the waiting queues and assigns calls to idle agents; see Chan et al. (2014) for a study of various routing policies. In this paper, we assume one waiting queue per call type and calls of the same type are handled *first-come, first-served* (FCFS). However, calls of different types may be handled in varying order depending on their priorities and on the availability of agents. With a single FCFS queue, future arrivals do not affect the waiting times (delays) of current customers, but this is not necessarily true in the multi-skill case. Predicting delay times in multi-skill call centers is much more difficult.

Waiting time has an important impact on the quality of service experienced by customers in several types of service systems, not only call centers. For example, statistical studies have been made to measure the negative effect of the delay experienced by travelers at an airport, customers at a restaurant, and patients at a hospital ((Taylor, 1994); (Dubé-Rioux et al., 1989); (Mowen et al., 1993)). Good predictors of the delay can be useful in these other types of systems as well. One particularity of call centers is that although the state of the queues and the customer abandonments are monitored continuously by the routing device, this information is accessible only from the managerial side, and is generally invisible to the customers, unless there is a mechanism of delay announcement. Some call centers may also propose the option of calling back the customer if the predicted waiting time is deemed too long. After learning the estimated delay, the customer may choose to stay in queue, abandon, or ask to be called back later Armony and Maglaras (2004). Providing such an option requires a good delay predictor.

1.2 Literature review

There has been a growing number of studies on delay time prediction and announcement for single-skill call centers, but for the general multi-skill setting there has been very limited work, and only on small or restricted models. For example, see Nakibly (2002) and more recently Senderovich et al. (2015). The research literature is roughly divided in two main categories. Several papers study the delay prediction along with the effects of delay announcement on the behavior of the customers; see, e.g., Whitt (1999a), Guo and Zipkin (2007), Armony et al. (2009), Jouini et al. (2009), and Jouini et al. (2011). A recurrent conclusion is that delay announcement helps reduce system congestion and increase service output, because impatient customers balk upon arrival (immediate abandonment), so fewer (patient) callers wait. A second branch of research focuses exclusively on delay predictions, without considering the impact on the behavior of customers. The works of Whitt (1999b), Ibrahim and Whitt (2008, 2009a, 2009b, 2010, 2011a, 2011b), and Ibrahim et al. (2015), as well as the present paper, fall into this category.

Two major families of delay predictors have been studied for call centers. Following the terminology of Ibrahim and Whitt (2009a), they are the *queue-length* (QL) based estimators and the *delay-history* (DH) based estimators. QL-based predictors essentially use the state of the queues and the system parameters to estimate the waiting time. For the single GI/M/s queue (with general inter-arrival time, exponential service time with rate μ, s servers, and no abandonments), it is well-known that the expected waiting time of a customer is $(n+1)/s\mu$ when all servers are busy and n other customers are waiting in queue in front of this customer (regardless of how long this customer has been waiting). For the GI/M/s+M model with abandonments (exponential patience time with rate ν), this expected waiting time is $\sum_{i=0}^{n}(s\mu+i\nu)^{-1}$ Whitt (1999b). Ibrahim and Whitt (2009a) and Ibrahim and Whitt (2011a) propose other variants of QL-based

predictors for a single queue. After completing our study, we found that Senderovich et al. (2015) worked, in parallel with us, on a multi-skill model that is more restrictive than ours. They design a QL-based predictor for queueing systems with multiple customer types, exponential distributions, a single agent group and a fixed priority routing policy (where customers with higher priority are always handled first). The customer types can have different exponential arrival rates, service rates and abandonment rates. In contrast to them, we consider more general queueing models with many agent groups, general distributions and nonspecific routing policy, but we do not propose a QL-based predictor.

The DH-based predictors, on the other hand, mainly use past delay information, instead of the state of the queues and system parameters, to estimate the waiting time of a new customer. Nakibly (2002), Armony et al. (2009), and Ibrahim and Whitt (2009b) propose simple heuristic predictors that return the delay times experienced by previous customers. As examples of DH-based predictors, one may return the waiting time of the *last customer to enter service* (LES), the customer at the *head of the line* (HOL), the *last customer to complete service* (LCS), or the *most recently arrived customer to complete service* (RCS).

Ibrahim and Whitt (2009a, 2011a, 2011b) compare QL- and DH-based predictors in the context of a single queue, by means of simulation and analytical comparisons. They conclude that QL-based predictors are generally more accurate and preferable to DH-based ones, if all the information that goes into the formula is available. Among the DH-based predictors mentioned above, LES and HOL are the best performers in their study, and their performance is comparable. These DH-based predictors also perform about the same as the QL-based predictors when the arrival rate and staffing are constant in time (or do not vary much), but their relative accuracy decreases when the time-variability of the arrival process increases. The DH-based predictors are attractive in practice because they require little and only observable information. They are easier to extend to the multi-skill setting and they may be more robust than QL-based predictors when dealing with unexpected events or unknown parameters. Unfortunately, the QL-based predictors mentioned above do not apply to multi-skill centers and are not easy to adapt, especially when the agents have different sets of skills.

1.3 Objective

This paper focuses on delay prediction in multi-skill call centers. This is an important problem that has barely been studied in the literature. Our main goals are: (i) to test the accuracy of the LES predictor in the multi-skill setting, and (ii) to propose new delay predictors that could compete with LES and eventually be more accurate. We do not directly consider the QL-based predictors since they do not extend naturally to the multi-skill context. They would need to take into account the skill sharing of the agents and the routing policy, and this appears complicated and difficult. Nevertheless, our new estimators use the queue lengths as input, in combination with other information.

Our proposed delay predictors are based on a heuristic approach that combines function approximation, machine learning, simulation, and ideas from single queue predictors. For each call type k, the predictor is a parameterized nonlinear function of the delay time of the last customer of type k who entered service (as in LES), the current queue length for call type k, and the queue lengths for all types $i \neq k$ for which there is an agent that can serve both types k and i. The parameter vector that defines the function is "optimized" (or *learned*) to minimize the mean square prediction error, based on either real historical data or data obtained from simulation with a model of the call center. In our experiments, we do the latter. We consider two types of predictor functions: (i) one defined by regression (smoothing) splines and (ii) the other defined by an artificial neural network. When a new customer enters queue k, the predictor function is evaluated, after observing the required inputs. The new predictors can be seen as extensions of LES, or partial combinations of LES and QL-based estimators. They require an extra "initialization" (learning) step.

1.4 Structure of the paper

The remainder is organized as follows. Section 2 describes the general multi-skill call center model. Section 3 presents the new proposed delay predictors, the input information they use, their parameters, and explains how these parameters are estimated (or learned). Numerical experiments with a single queue and two N-

model call centers (with two call types and two agent groups) are reported in Section 4. Finally, concluding remarks are given in Section 5.

2 Call center model

We consider a general multi-skill call center model in which each customer is categorized into one of the K possible call types, and agents are divided into G groups. An agent of group $g \in \{1, \ldots, G\}$ has the skill set $\mathcal{S}_g \subseteq \{1, \ldots, K\}$ which defines the set of call types she can serve. The opening hours of the call center are divided into P time periods of equal length. Since we use simulation and do not rely on any queueing formulas, the arrival process, service times, and patience times can have very general distributions. Their choice does not affect the mechanism of our delay predictors. In our numerical examples, we shall assume that for call type k, the arrival process is a Poisson process with a constant rate $\lambda_{k,p}$ over each period p, so the vector of arrival rates over the P periods is $\lambda_k = (\lambda_{k,1}, \ldots, \lambda_{k,P})$. These arrival processes are assumed independent across call types. For type k, the service times are assumed exponential with mean $1/\mu_k$, the patience times are exponential with mean $1/\nu_k$, and all these random variables are independent. A customer abandons the queue once her waiting time exceeds her patience time. We do not model retrials. There is no service preemption, which means an agent cannot interrupt a call in service. Let $s_g = (s_{g,1}, \ldots, s_{g,P})$ be the staffing vector of group g, where $s_{g,p}$ is the number of agents from that group working in period p. Agents in the same group are considered homogeneous.

There is one waiting queue per call type. A new call of type k is placed at the end of queue k if, upon its arrival, there is no idle agent with the skill to serve it. The router assigns calls to idle agents according to a routing policy, which we leave open for now. It will be specified in our examples. There is no callback option and we assume that the delay predictions have no influence on the behavior of the customers or on the operations of the call center.

3 Delay predictors

3.1 Approximating the conditional expectation of the delay

The waiting time $W > 0$ of a given customer that enters a queue and waits until her service begins is a random variable whose distribution depends on the type k of this customer and on the state of the system when this customer arrives. As a simplistic predictor of W, one may just take the global average waiting time for type k customers, which can be estimated by simulation (we assume that a simulation model of the system is available). This predictor is the unconditional expectation of W, $\mathbb{E}_k[W]$, when we take only k as input and we look at no other information. It is called the *No-Information* (NI) predictor Ibrahim and Whitt (2009b). (Note that we have defined W only for a customer that enters the queue and waits until being served, so the expectation is always conditional on this.)

To make better predictions, the general idea is to observe the state of the system when the customer enters the queue and return an estimate of the expectation of W conditional on that state (given k). In practice, we will select some information $\boldsymbol{x}$ from the system's state, and compute an approximation of the conditional expectation $\mathbb{E}_k[W \mid \boldsymbol{x}]$, which depends on k. This approximation is defined by a *predictor function* $F_{k,\theta}(\boldsymbol{x})$ of the observed (input) information vector $\boldsymbol{x}$, where θ is a parameter vector estimated (or learned) previously.

In this paper, we take $\boldsymbol{x} = (t, q, \boldsymbol{r})$ where t is the waiting time of the last call of type k to have entered service, q is the number of calls already in queue k, and $\boldsymbol{r}$ is a vector that contains the size of each queue $j \neq k$ such that there is at least one agent with both skills k and j.

We consider two ways of constructing the functions $F_{k,\theta}$. In the first one, each function is a smoothing (least-squares regression) cubic spline which is additive in the input variables. In the second one, the function is defined by a deep feedforward multilayer artificial neural network (ANN). They are described below. We will compare them with LES and also with the simplistic NI predictor that always returns the average waiting time as a prediction (it corresponds to taking $\boldsymbol{x}$ as empty).

The predictors are optimized to minimize the *mean squared errors* (MSEs) of predictions. If $E = F_{k,\theta}(\boldsymbol{x})$ is the predicted delay for a "random" customer of type k that opts to wait and W is its realized waiting time, the MSE for type k calls is defined as

$$\mathrm{MSE}_k = \mathbb{E}[(W - E)^2].$$

We cannot compute this MSE exactly, so we estimate it by its empirical counterpart (and consistent estimator), the *average squared error* (ASE) of the predictions. We use simulation to compute the ASE. Let C_k be the number of observed calls of type k that experienced strictly positive wait time before they received service in the simulation. Suppose the cth call among those has a realized waiting time of $W_{k,c} > 0$ and a predicted delay time of $E_{k,c}$ upon arrival, then the ASE for type k is

$$\mathrm{ASE}_k = \frac{1}{C_k} \sum_{c=1}^{C_k} (W_{k,c} - E_{k,c})^2. \tag{1}$$

Note that our definition of the ASE differs from Ibrahim and Whitt (2009b), who also include virtual delays for customers who have abandoned.

To estimate (or learn) the parameter vector θ, we use a learning data set generated by simulation. Let $\boldsymbol{x}_{k,c}$ represent the information vector $\boldsymbol{x}$ when the cth call (among C_k) of type k joins the waiting queue. We would like to select θ to minimize (w.r.t. θ) the ASE_k as defined in (1) with $E_{k,c} = F_{k,\theta}(\boldsymbol{x}_{k,c})$. However, other factors may also enter the objective function; for example the smoothness of the predictor function in the case of the splines (see below).

When we compare the accuracy of the predictors in our numerical experiments, we generate a separate and independent (out-of-sample) set of observations by simulation. Instead of the ASE, we report a normalized version of the ASE, called the *root relative average squared errors* (RRASE), measured on this new set of data. The RRASE for type k is

$$\mathrm{RRASE}_k = \frac{\sqrt{\mathrm{ASE}_k}}{(1/C_k) \sum_{c=1}^{C_k} W_{k,c}}.$$

The RRASE_k's of the predictors are measured on the same out-of-sample set of data, so the $W_{k,c}$'s and C_k's are identical across predictors. If we replace the ASE by the RRASE in the estimation phase of θ, the optimal solution will not change, because the square-root operator is monotone increasing and the denominator is a positive constant.

3.2 Regression (smoothing) splines (RS)

Splines provide a well-known and powerful class of approximation methods for general smooth functions de Boor (1978). Here we use smoothing cubic splines, for which the parameters are estimated by least-squares regression together with a penalty term on the function variation, to promote smoother functions. We also restrict ourselves to additive splines, which can be written as a sum of one-dimensional functions. That is, if the information vector is written as $\boldsymbol{x} = (x_1, \ldots, x_D)$, the additive spline predictor can be written as

$$F_{k,\theta}(\boldsymbol{x}) = \sum_{d=1}^{D} f_d(x_d),$$

where each f_d is a one dimensional cubic spline. The parameters of all these spline functions f_d form the vector θ. These parameters must satisfy the constraints that the successive pieces of the spline (which are cubic polynomials) have their first and second derivatives equal at the border. To estimate the parameters, we use the function `gam` implemented in the package `mgcv` for the R statistical software Wood (2006), R Core Team (2014). The number of knot points and the smoothing factors are chosen automatically by the package, based on the data.

3.3 Artificial neural networks (ANNs)

ANNs are another very popular and effective way to approximate complicated high-dimensional functions. One recent trend is *deep learning*, which refers to the use of ANNs with several layers of neurons Bengio

et al. (2012), LeCun et al. (2015). We adopt this technology here. To train the ANN (i.e., to estimate a good parameter vector θ), we use the state-of-the-art software Pylearn2 Goodfellow et al. (2013). We have selected a deep feedforward ANN in which the outputs of nodes at layer l are the inputs of every node at the next layer $l+1$. A typical ANN has one input layer, one output layer and several hidden layers. There currently exists no method to determine the optimal number of hidden layers or number of nodes in them. In practice, these numbers are usually chosen, by trial and error, after some preliminary runs. In our numerical examples, we use five layers: one input layer, three hidden layers and one output layer. The number of nodes at the input layer is equal to the number of elements in the parameter vector $\boldsymbol{x}$, and the output layer has only one node which returns the estimated delay. The number of nodes in a hidden layer depends on the size of the call center; this number is specified in the numerical section. For each hidden node, we use a *rectifier* activation function $h(\boldsymbol{z}) = \max(0, b + \boldsymbol{w} \cdot \boldsymbol{z})$, in which $\boldsymbol{z}$ is the vector of inputs for the node, while the intercept b and the vector of coefficients $\boldsymbol{w}$ are parameters learned by training. The (large) vector θ is the set of all these parameters b and $\boldsymbol{w}$, over all nodes. This type of activation function has been proposed recently Glorot et al. (2011), and it is thought to represent more faithfully the biological mechanism of a neuron than the conventional sigmoid and hyperbolic tangent functions. The parameters are learned by a back-propagation algorithm that uses a gradient descent method Bishop (2006).

These ANNs are very powerful, but one drawback is that they require larger training samples and their training can be much more time-consuming than other regression techniques such as splines. To speedup the learning, we use data aggregation, as follows. The idea is to aggregate observations whose values of $\boldsymbol{x}$ are almost the same, and replace them by a single observation $(\boldsymbol{x}', w')$, where w' is the average waiting time for the observations that have been aggregated. This new aggregated observation will have a weight proportional to the number of original observations that were aggregated into it. To form the groups that are aggregated, we first regroup all observations $\boldsymbol{x}$ having the same pair $(q, \boldsymbol{r})$, then divide each of those groups into 20 subgroups of approximately equal size according to the value of t. For this, we use the 5%, 10%, 15%, ..., quantiles with respect to t as separators to make the subgroups, then aggregate each subgroup. The aggregated observation is $(\boldsymbol{x}', w') = ((t', q, \boldsymbol{r}), w')$, where w' is the average delay for the subgroup and t' is the midpoint of the interval between the corresponding quantiles. The set of aggregated observations is used as the new training data set.

4 Numerical experiments

We compare the performance and accuracy of the LES, RS and ANN predictors by simulating small models of call centers. We start with the classical (single) M/M/s queue as a formality, to see if our RS and ANN predictors are competitive with the best predictors available for this simple model. Next, we test our predictors on two N-models of call centers, one with small queues and short waits and the other one with long queues and long waits.

As explained in Section 3.1, the predictors are optimized to minimize the ASE, but we report the normalized root value, the RRASE. To train the RS and ANN predictors, we generate call observations by simulation. To evaluate the M/M/s in steady-state, we simulate one long run of 600,000 hours and the observations from the first 200,000 hours (warmup time) are discarded. For the N-models, we simulate 100 independent days to generate the observations. For the ANN, we take 80% of the observations as the training data and the remaining 20% as the test data used to measure the fitting, which is used to select the best parameter θ among those found in the training. To compare the RRASE of the different predictors, we generate an independent set of observations by running another simulation of the same length, for each model, and we use this same set to compute the RRASE for all the predictors, so the predictors are effectively compared on the same data. All times are measured in seconds. To give a rough idea on the "cost" of learning (from the same data set), RS requires 1 minute or less, while ANN needs around 1 hour for M/M/s and 2 hours for the N-models. The prediction step (once learned) is very fast with both methods.

4.1 Experiments with an M/M/s model

We consider a single queue with a Poisson arrival process with rate $\lambda = 50$ calls per hour, exponential service times with rate $\mu = 2$, and $s = 26$ agents. There are no abandonments. In the simulations used to evaluate

the predictors, we found an average queue length of 19.5 customers, a delay probability of 78.2%, and a conditional average waiting time of 1795 seconds for the customers that entered the queue. In steady-state over an infinite horizon, for comparison, the average queue length is 19.6 customers, and the conditional average waiting time for customers that wait is 1800 seconds.

The NI predictor returns a constant which is the expected waiting time of a customer that enters the queue. We include the QL predictor in this comparison, because it is applicable in this single queue example. For the ANN, we use 8 nodes in each of the 3 hidden layers. We test a variant of RS where the input is only q (the number of customers already in queue). We also test a variant of ANN to which we give the same inputs as the QL predictor, that is, the queue length q, and the constant inputs μ and s. These variants are named RS(q) and ANN(q, μ, s), respectively.

Table 1 shows the RRASE obtained for each predictor. As expected, QL has the best result since it is an optimal delay predictor for a M/M/s queue in steady-state. RS and ANN (including their variants) arrive closely behind QL, followed by LES. The good results of RS(q) and ANN(q, μ, s) show that the QL predictor function can be learned with machine learning methods. NI performs much worse, without surprise, and this confirms that using state-dependent predictors is worthwhile.

Table 1: The RRASE on the M/M/s model.

Result	NI	LES	QL	RS	RS(q)	ANN	ANN(q, μ, s)
RRASE	0.956	0.266	0.244	0.256	0.246	0.258	0.248

4.2 Experiments with the N-model

We now consider an N-model, with 2 call types and 2 agent groups, where the groups have skill sets $\mathcal{S}_1 = \{1\}$ and $\mathcal{S}_2 = \{1, 2\}$. Group 1 can serve only calls of type 1, and group 2 can serve all calls. A day is divided into $P = 10$ periods of 1 hour. The arrival processes are Poisson with constant rate in each period. All service times and patience times are exponential and independent. We use a *priority routing* policy Chan et al. (2014) that works as follows. For a call of type 1, the router will first try to match it with an idle agent of group 1. If there is no such free agent, then the router will try to assign it to an idle agent of group 2. Agents of group 2 always give first priority to calls of type 2, even if some calls of type 1 have waited longer. Thus, calls of the same type are first-come first-served, but calls of different types may be served in different orders.

We compare the LES, RS and ANN predictors. We do not include QL because it is not directly applicable (there is no formula) in the multi-skill context. The input for RS and ANN is $\boldsymbol{x} = (t, q, \boldsymbol{r})$ for both call types, where $\boldsymbol{r}$ is a vector of length 1. For the ANN, we found that 180 nodes per hidden layer was sufficient. We also consider variants of RS and ANN where the size of the secondary queue (the input $\boldsymbol{r}$) is removed from the predictor input. We name these variants RS(t, q) and ANN(t, q). In each case, we report the RRASE for call type 1, call type 2, and aggregated over the two types.

4.2.1 An N-model with short queues

Our first N-model example is a case with short queues. For calls of type 1, the vector of arrival rates (per period) are $\lambda_1 = (16, 20, 28, 30, 35, 45, 40, 30, 20, 15)$ per hour, the mean service time is $\mu_1^{-1} = 20$ minutes, and the mean patience time is $\nu_1^{-1} = 25$ minutes. For type 2, the arrival rates are $\lambda_2 = (20, 32, 40, 50, 60, 50, 40, 35, 30, 20)$ per hour, the mean service time is $\mu_2^{-1} = 10$ minutes, and the mean patience time is $\nu_2^{-1} = 20$ minutes. The staffing vectors (per period) are $s_1 = (3, 5, 8, 8, 9, 10, 9, 6, 5, 5)$ and $s_2 = (4, 6, 8, 10, 9, 9, 8, 8, 6, 5)$.

The performance measures aggregated over all periods are as follows. For call type 1, there is an average of 1.6 customers in queue, the delay probability is 61%, the abandonment ratio is 14%, the average waiting time is 211 seconds (for all customers), and the average waiting time of customers who waited and were served was 354 seconds. For type 2, these average measures are respectively: 2.9 customers in queue, 79% delayed, 12% abandonments, 193 seconds of wait, and 248 seconds of wait. The skill sharing is very present:

80% of served calls of type 1 were answered by agents of group 1, whereas the other 20% were answered by agents of group 2. Although the global average waiting times (over all customers) differ by less than 20 seconds between the 2 call types, the average waiting times for those who waited and were served are quite different for the two types. Figure 1 shows the waiting time distribution of customers who waited and were served, for each type. Type 1 has a longer tail and obviously a larger variance.

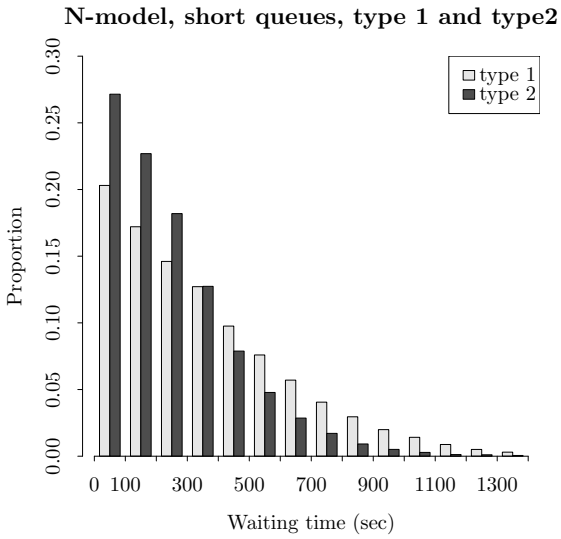

Figure 1: N-model with short queues: Distribution of the waiting time of customers who waited and were served, for each type.

Table 2 reports the RRASEs. It shows that our RS and ANN predictors and their variants have a RRASE that is around 25% less than that of the LES predictor, for all call types. Thus, including or not the parameter $\boldsymbol{r}$ (the length of the secondary queue) into the input $\boldsymbol{x}$ has little effect on the accuracy of RS and ANN. RS performs slightly better than ANN, for both the originals and the variants. Figure 2 gives a histogram of the prediction error for each method (excluding the variants). We see that the LES predictor has a much wider error distribution (large errors are more frequent), whereas RS and ANN have very similar error distributions, for both call types.

Table 2: The RRASE for the N-model with short queues.

Call type	LES	RS	RS(t,q)	ANN	ANN(t,q)
Type 1	0.871	0.603	0.612	0.612	0.621
Type 2	0.850	0.597	0.600	0.630	0.611
Overall	0.874	0.609	0.615	0.618	0.625

4.2.2 An N-model with long queues

Our second N-model example will have long queues. We take larger arrival rates and patience times than in the previous case, while keeping the staffing almost unchanged. For call type 1, the arrival rates are $\lambda_1 = (25, 34, 43, 48, 51, 57, 42, 34, 22, 18)$ per hour, the mean service time is $\mu_1^{-1} = 21$ minutes, and the mean patience time is $\nu_1^{-1} = 46.7$ minutes. For type 2, the arrival rates are $\lambda_2 = (26, 40, 47, 59, 68, 59, 48, 43, 39, 29)$ per hour, the mean service time is $\mu_2^{-1} = 11$ minutes, and the mean patience time is $\nu_2^{-1} = 30$ minutes. The staffing vectors are $s_1 = (4, 6, 9, 10, 9, 9, 9, 8, 5, 5)$ and $s_2 = (4, 7, 9, 10, 9, 8, 7, 8, 6, 5)$.

The performance measures aggregated over all periods are as follows. For call type 1, we find an average of 9.7 customers in queue, a delay probability of 94%, an abandonment ratio of 33%, an average waiting time of 938 seconds for all calls, and an average wait time of 1151 seconds for the calls that entered the

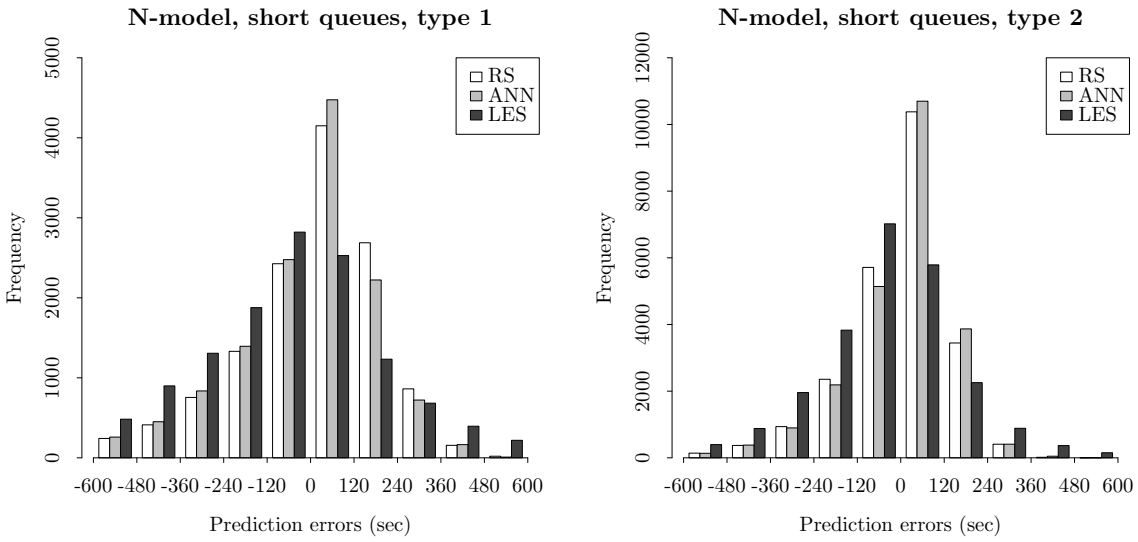

Figure 2: N-model with short queues: Distribution of the prediction errors (estimate minus real delay) for type 1 and type 2.

queue and were served. For type 2, these measures are 5.5 customers, 97% delayed, 23% abandonments, 426 seconds, and 465 seconds, respectively. In this example, 88% of the served calls of type 1 were answered by group 1, and the other 12% were answered by group 2. Figure 3 shows that the waiting time distributions for customers who waited and got served are very different between call types 1 and 2.

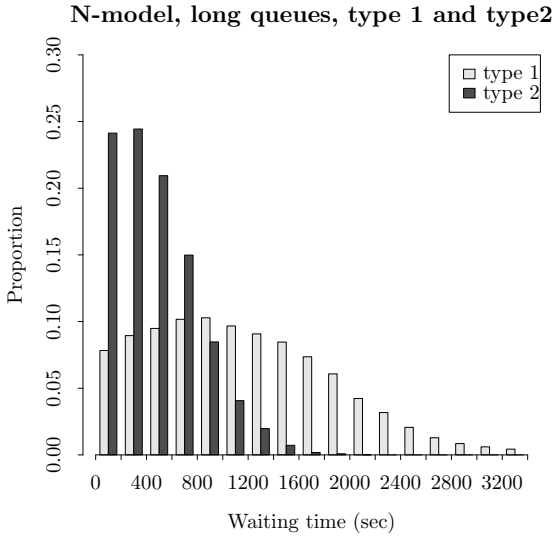

Figure 3: N-model with long queues: Distribution of the waiting time for customers who waited and got served.

By looking at the RRASEs in Table 3, we find that the comparison between the predictors is very similar to what we saw in the previous example with short queues. LES performs much more poorly than all our new methods; its RRASE is larger by 13% to 19%. Again, RS is a little better than ANN. Adding the parameter $\boldsymbol{r}$ to the input $\boldsymbol{x}$ improves slightly the accuracy of RS, but it has no significant impact on ANN. Figure 4 also illustrates the largest variance of the prediction error with LES.

Table 3: The RRASE for the N-model with long queues.

Call type	LES	RS	RS(t,q)	ANN	ANN(t,q)
Type 1	0.499	0.364	0.369	0.370	0.370
Type 2	0.629	0.443	0.446	0.446	0.446
Overall	0.581	0.418	0.422	0.423	0.423

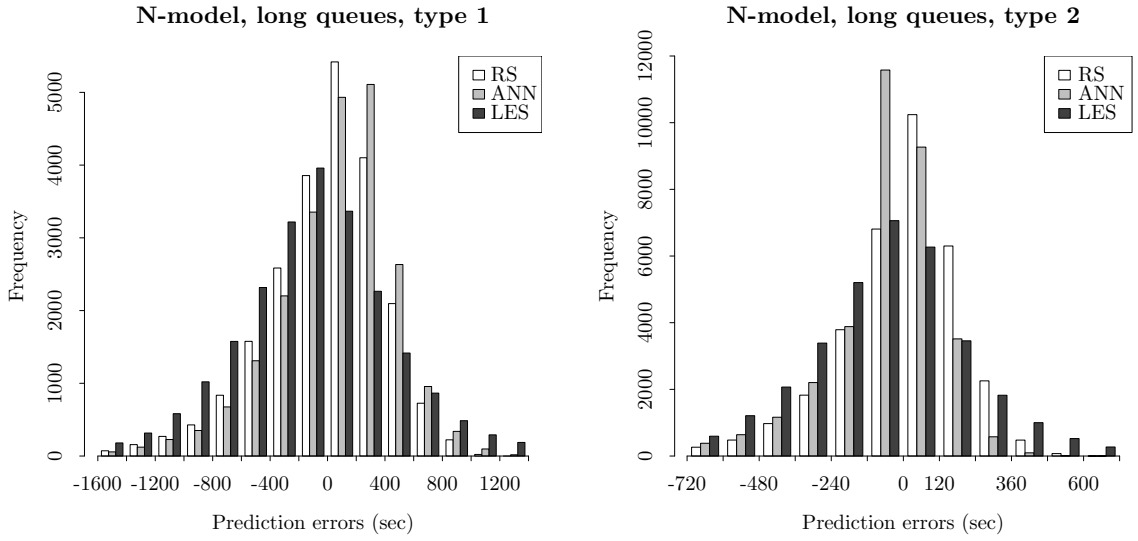

Figure 4: N-model with long queues: Distribution of the prediction error (estimate minus real delay) for call types 1 and 2.

5 Conclusion

We have introduced new delay predictors for multi-skill call centers, built by using RS and ANN. In our numerical experiments, both RS and ANN were much more accurate than the best existing strategy we know from the literature, which returns the waiting time of the last customer of same type that has started service. The accuracies between RS and ANN are very similar, but RS can be trained much quicker than ANN for the considered examples. We will compare these predictors for larger models in a future work. Our predictors return a point estimate of the waiting time based on an approximation of the conditional expectation of the waiting time conditional on the current state of the system when the customer enters the queue. This current state is represented by the information (input) vector $\boldsymbol{x}$. In our on-going and future work, we want to develop effective methods to predict and announce not only a point estimate of the waiting time (an estimate of the expectation), but an estimate of the entire conditional distribution of the delay, or at least some of its quantiles.

References

M. Armony and C. Maglaras. On customer contact centers with a call-back option: Customer decisions, routing rules, and system design. Operations Research, 52(2):271–292, 2004.

M. Armony, N. Shimkin, and W. Whitt. The impact of delay announcements in many-server queues with abandonments. Operations Research, 57(1):66–81, 2009.

Y. Bengio, A.C. Courville, and P. Vincent. Unsupervised feature learning and deep learning: A review and new perspectives. CoRR, abs/1206.5538, 2012. http://arxiv.org/abs/1206.5538.

C.M. Bishop. Pattern Recognition and Machine Learning. Springer, 2006.

W. Chan, G. Koole, and P. L'Ecuyer. Dynamic call center routing policies using call waiting and agent idle times. Manufacturing & Service Operations Management, 16(4):544–560, 2014.

C. de Boor. A Practical Guide to Splines. Number 27 in Applied Mathematical Sciences Series. Springer-Verlag, New York, 1978.

L. Dubé-Rioux, B.H. Schmitt, and F. Leclerc. Consumers' reactions to waiting: When delays affect the perception of service quality. Advances in Consumer Research, 16:59–63, 1989.

N. Gans, G. Koole, and A. Mandelbaum. Telephone call centers: Tutorial, review, and research prospects. Manufacturing and Service Operations Management, 5:79–141, 2003.

X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In G.J. Gordon and D.B. Dunson, editors, Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS-11), volume 15, pages 315–323. Journal of Machine Learning Research – Workshop and Conference Proceedings, 2011. http://www.jmlr.org/proceedings/papers/v15/glorot11a/glorot11a.pdf.

I.J. Goodfellow, D. Warde-Farley, P. Lamblin, V. Dumoulin, M. Mirza, R. Pascanu, J. Bergstra, F. Bastien, and Y. Bengio. Pylearn2: A machine learning research library. arXiv preprint arXiv:1308.4214, 2013. http://arxiv.org/abs/1308.4214.

P. Guo and P. Zipkin. Analysis and comparison of queues with different levels of delay information. Management Science, 53(6):962–970, 2007.

R. Ibrahim and W. Whitt. Real-time delay estimation in call centers. In Proceedings of the 2008 Winter Simulation Conference, pages 2876–2883. IEEE Press, 2008.

R. Ibrahim and W. Whitt. Real-time delay estimation in overloaded multiserver queues with abandonments. Management Science, 55(10):1729–1742, 2009a.

R. Ibrahim and W. Whitt. Real-time delay estimation based on delay history. Manufacturing and Services Operations Management, 11:397–415, 2009b.

R. Ibrahim and W. Whitt. Delay predictors for customer service systems with time-varying parameters. In Proceedings of the 2010 Winter Simulation Conference, pages 2375–2386. IEEE Press, 2010.

R. Ibrahim and W. Whitt. Wait-time predictors for customer service systems with time-varying demand and capacity. Operations Research, 59(5):1106–1118, 2011a.

R. Ibrahim and W. Whitt. Real-time delay estimation based on delay history in many-server service systems with time-varying arrivals. Production and Operations Management, 20(5):654–667, 2011b.

R. Ibrahim, M. Armony, and A. Bassamboo. Does the past predict the future? The case of delay announcements in service systems. 2015. Manuscript.

O. Jouini, Y. Dallery, and Z. Aksin. Queueing models for full-flexible multi-class call centers with real-time anticipated delays. International Journal of Production Economics, 120:389–399, 2009.

O. Jouini, Y. Dallery, and Z. Aksin. Call centers with delay information: Models and insights. Manufacturing and Service Operations Management, 13(4):534–548, 2011.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. Nature, 521:436–444, 2015. http://dx.doi.org/10.1038/nature14539.

J.C. Mowen, J.W. Licata, and J. McPhail. Waiting in the emergency room: How to improve patient satisfaction. Journal of Health Care Marketing, 16(2):26–33, 1993.

E. Nakibly. Predicting waiting times in telephone service systems. Master's thesis, Technion, Haifa, Israel, 2002.

R Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, 2014. http://www.R-project.org.

A. Senderovich, M. Weidlich, A. Gal, and A. Mandelbaum. Queue mining for delay prediction in multi-class service processes. Information Systems, 53:278–295, 2015. http://dx.doi.org/10.1016/j.is.2015.03.010.

S. Taylor. Waiting for service: The relationship between delays and evaluations of service. Journal of Marketing, 58 (2):56–69, 1994.

W. Whitt. Improving service by informing customers about anticipated delays. Management Science, 45(2):192–207, 1999a.

W. Whitt. Predicting queueing delays. Management Science, 45(6):870–888, 1999b.

S.N. Wood. Generalized Additive Models: An Introduction with R. Texts in Statistical Science Series. Chapman & Hall/CRC, Boca Raton, FL, 2006. ISBN 978-1-58488-474-3; 1-58488-474-6.