Machine Learning and the Traveling Repairman

Theja Tulabandhula THEJA@MIT.EDU
Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Cynthia Rudin RUDIN@MIT.EDU
Sloan School of Management and Operations Research Center
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Patrick Jaillet JAILLET@MIT.EDU
Laboratory for Information and Decision Systems, Department of Electrical Engineering and Computer Science, and Operations Research Center
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Abstract

The goal of the *Machine Learning and Traveling Repairman Problem* (ML&TRP) is to determine a route for a "repair crew," which repairs nodes on a graph. The repair crew aims to minimize the cost of failures at the nodes, but as in many real situations, the failure probabilities are not known and must be estimated. We introduce two formulations for the ML&TRP, where the first formulation is sequential: failure probabilities are estimated at each node, and then a weighted version of the traveling repairman problem is used to construct the route from the failure cost. We develop two models for the failure cost, based on whether repeat failures are considered, or only the first failure on a node. Our second formulation is a multi-objective learning problem for ranking on graphs. Here, we are estimating failure probabilities simultaneously with determining the graph traversal route; the choice of route influences the estimated failure probabilities. This is in accordance with a prior belief that probabilities that cannot be well-estimated will generally be low. It also agrees with a managerial goal of finding a scenario where the data can plausibly support choosing a route that has a low operational cost.

1. Introduction

We consider the problem of routing an agent ("repair crew") on a graph, where each node in the graph has some probability of failure. The probabilities are not known and must be estimated from past failure data. Ideally, the nodes that are most prone to failure should be repaired first, but if those nodes are far away from each other, the extra time spent traveling between nodes might actually increase the chance of failures occurring at nodes that have not yet been repaired. In that sense, it is better to construct the route to minimize the cost of the failures, taking into account the travel time between nodes and also the (estimated) failure probabilities at each of the nodes. We call this problem the *machine learning and traveling repairman problem* (ML&TRP), and in this work, we present

systematic approaches to formulating and solving this problem. There are many possible applications of the ML&TRP, including the scheduling of safety inspections or repair work for the electrical grid, oil rigs, underground mining, machines in a factory, or airplanes. Another example is to route delivery trucks that carry items that may be damaged if the items are in the vehicle too long (e.g., ice cream or other groceries).

We present two formulations for the ML&TRP. The first formulation is *sequential*: the failure probabilities are estimated, and then the probabilities determine a graph traversal cost for each possible route. The route that minimizes the graph traversal cost is then determined by solving a weighted traveling repairman problem (TRP), also called a minimum latency problem, or more generally, a time-dependent traveling salesman problem (see for instance, Picard and Queyranne, 1978). The second formulation computes the probabilities and the route *simultaneously*, by minimizing an objective with two terms. The first term is a training error term used for estimating probabilities and the second term is the graph traversal cost. This means that estimated failure probabilities are chosen together with knowledge of the graph traversal cost. The graph traversal cost acts as a regularization term, and has a tendency to lower probability estimates and promote generalization; this is in accordance with a prior belief that the probabilities will be low if they cannot be well-estimated. The algorithm will thus prefer routes where the first nodes visited actually have higher failure probabilities. Another reason to incorporate a prior belief that the graph traversal cost will be low is managerial. A company might wish to know whether it is at all possible that a low-cost route can be designed, where the operational costs are realistically supported by the data. Among all reasonable probability models, our second formulation will find one that corresponds to the lowest-cost route. This type of formulation is optimistic; it provides the best possible (but still reasonable) scenario described by the data.

We design the graph traversal cost in two ways, where either traversal cost can be used for either the sequential or the simultaneous formulations. The first graph traversal cost is the expected cost of failures, where multiple expected failures at the same node each contribute to the cost. The second graph traversal cost is inspired by (but not equal to) the expected cost of the first failure at each node. The first cost applies when the failure probability of a node does not change until it is visited by the crew, regardless of whether a failure already occurred at that node, and the second cost applies when the node is completely repaired after the first failure, or when it is visited by the repair crew, whichever comes first. Thus, the choice of cost function should depend on the types of failures and repairs considered in the application.

In the second formulation, regularizing by the graph traversal cost limits the complexity of the hypothesis space used for the probabilistic model. This means that the graph traversal cost term may assist with generalization, that is, the probabilistic model's ability to predict well on data drawn from the same distribution. In this work we present a generalization bound showing how a limitation on the graph traversal cost might lead to a more accurate model for failure probabilities.

The ML&TRP problem does not fall under the umbrella of semi-supervised learning since the incorporation of unlabeled data is used for determining the route cost, and is not used to provide additional *distributional* information (see for instance, Chapelle et al., 2006). Also our approach to regularization along the route is entirely different from work

on graph regularization (Agarwal, 2006; Belkin et al., 2006; Zhou et al., 2004), which does not concern traversal routes. In that work, there is an assumption that probabilities will be smooth along the graph. This assumption is not true for many applications; see for instance, the power grid application discussed below.

We will discuss a motivating example for the ML&TRP in the remainder of the introduction. In Section 2 we will outline the two general formulations, and provide the two methods for computing the route cost. In Section 3 we provide mixed-integer nonlinear programs (MINLP's) for solving the ML&TRP. Section 4 gives relevant illustrations, along with some experiments on data from the NYC power grid. Section 5 contains the theoretical generalization result, and in Section 6 we discuss related literature.

Motivation

One particularly motivating application for the ML&TRP is smart grid maintenance. Since 2004, many power utility companies are implementing new inspection and repair programs for preemptive maintenance, whereas in the past, all repair work was done reactively (Urbina, 2004). An example of this is *vented manhole cover replacement programs*, where each manhole in a city is replaced with a vented cover that allows gases to escape, mitigating the possibility and effects of serious events including fires and explosions.

New York City's power company Con Edison, which has such a replacement program, services tens of thousands of manholes in each borough, and it is not sensible for a repair crew to travel across the city and back again for each cover replacement. The scheduling of manhole inspection and repair work in Manhattan, Brooklyn and the Bronx is assisted by a machine learning model that estimates the probability of failure for each manhole within a given year (Rudin et al., 2010). Features for the model are derived from physical characteristics of the manhole (e.g., number of cables entering the manhole), and features derived from its history of involvement in past events. Repeat failures (serious and non-serious events) can occur on the same manhole. That said, failures are rare events, and it is not easy to accurately estimate the probability that a given manhole will fail within a given period of time. The current model for estimating failures does not take into account the route of the repair crew that replaces the covers. This leaves open the possibility that, for this domain and for many other domains, estimating the failure probabilities with knowledge of the route optimization procedure could lead to an improvement in repair operations.

The features for the NYC machine learning models are recomputed periodically, but not often, due to the expense of processing the raw data, and the fact that these probabilities change very slowly with time. Because of this, the route must be determined before the work starts. Also, the probabilities are not smooth along the graph, as discussed by Rudin et al. (2010). In initial attempts to model these probabilities on the Manhattan power grid, estimates were smooth geographically, and this type of model did not perform nearly as well as a more targeted model. In Manhattan it is very common to have relatively vulnerable manholes right next to manholes that are not vulnerable.

The limited resources for inspection and repair of manholes should generally be designated to the most vulnerable manholes. With uncertainty in many of the probability estimates, if we are not careful, it is possible that most of these resources will be spent in dealing with outliers whose probabilities are overestimated. Our second ML&TRP for-

mulation aims to prevent this from happening. At the same time, that formulation may be able to find a solution that is supported by the data, but also meets operational cost requirements or targets.

2. ML&TRP Formulations

We first provide some notation. We are given two sets of instances, $\{x_i\}_{i=1}^m$, $\{\tilde{x}_i\}_{i=1}^M$, with $x_i \in \mathcal{X}$, $\tilde{x}_i \in \mathcal{X}$ that are feature vectors with $\mathcal{X} \subset \mathbb{R}^d$. Let the x_i^j indicate the j-th coordinate of the feature vector x_i. For the first set of instances, we are also given labels $\{y_i\}_{i=1}^m$, $y_i \in \{-1, 1\}$. These instances and their labels are the set of training examples. The other instances $\{\tilde{x}_i\}_{i=1}^M$are unlabeled data that are each associated with a node on a graph G, where the distance between nodes i and j, $d_{i,j} \in \mathbb{R}_+$ serves as the weight on the edge connecting them. The graph defined by them is complete. A route on G is represented by a permutation π of the node indices $1, \ldots, M$. Let Π be the set of all permutations of $\{1, ..., M\}$. A set of failure probabilities will be estimated at nodes and these estimates will be derived from a function of the form f_λ, where $f_\lambda : \mathcal{X} \to \mathbb{R}$, $f_\lambda \in \mathcal{F}$. The class of possible function models $\mathcal{F}$ is chosen to be the set of linear combinations of the feature coordinates:

$$\mathcal{F} := \{f : f(x) = \lambda \cdot x \text{ for some } \lambda \in \mathbb{R}^d \text{ such that } ||\lambda||_2 \leq M_1\}. \tag{1}$$

where M_1 is a fixed positive real number. We can easily make the function class more expressive by using affine functions (such as $g(x) = \lambda \cdot x + d$). But since $g(x)$, like $f(x)$, can be written as an inner product of a new parameter $[\lambda \;\; d]$ and a new feature $[x \;\; 1]$, we can append our feature vectors to form new vectors where the last coordinate of x is always 1 and use function class $\mathcal{F}$ as described.

For the smart grid maintenance example, the training instances might be manholes represented by features derived from data prior to 2010, and the labels might encode whether the manhole experienced a serious event within 2010. The test instances might represent features derived from data prior to 2011, and the goal is to create a repair route that minimizes the cost of repairs in 2011.

The sequential formulation for the ML&TRP follows two steps. The TrainingError and GraphTraversalCost objectives will be defined shortly.

Sequential Formulation

Step 1. Compute the scores $f_\lambda^*(\tilde{x}_i)$:

$$f_\lambda^* \;\; \in \;\; \text{argmin}_{f_\lambda \in \mathcal{F}} \text{TrainingError}(f_\lambda, \{x_i, y_i\}_{i=1}^m).$$

Step 2. Compute a route corresponding to the scores:

$$\pi^* \in \text{argmin}_{\pi \in \Pi} \text{GraphTraversalCost}(\pi, f_\lambda^*, \{\tilde{x}_i\}_{i=1}^M, \{d_{i,j}\}_{i,j=1}^M).$$

The result $\pi^* \in \Pi$ is the route used for the repair crew. In the first step, a transformation of $f_\lambda^*(x)$ yields an estimate of probability of failure $P(y = 1|x)$. Here, $\{y = 1\}$ is the event that a failure occurs on any given day or time step. Let the distances be scaled appropriately so that a unit of distance is traversed in a unit of time. We assume that the probability of

failure is the same at each time step until something happens (either the crew visits, or a failure occurs), and that x does not change over the time that the route is being traversed. To ensure that these probabilities are in agreement with past observations, we choose $f^*_\lambda(x)$ to minimize a training error in Step 1. In the second step, the route is chosen to minimize a weighted TRP cost based on those estimated probabilities.

The sequential formulation is easier to solve than the simultaneous formulation outlined below.

Simultaneous Formulation

Step 1. Compute the scores $f^*_\lambda(\tilde{x}_i)$:

$$\begin{aligned} f^*_\lambda \quad \in \quad & \text{argmin}_{f_\lambda \in \mathcal{F}} \Big[\text{TrainingError}(f_\lambda, \{x_i, y_i\}_{i=1}^m) \\ & + C_1 \min_{\pi \in \Pi} \text{GraphTraversalCost}\left(\pi, f_\lambda, \{\tilde{x}_i\}_{i=1}^M, \{d_{i,j}\}_{i,j=1}^M\right) \Big]. \end{aligned}$$

Step 2. Compute a route corresponding to the scores:

$$\pi^* \in \text{argmin}_{\pi \in \Pi} \text{GraphTraversalCost}(\pi, f^*_\lambda, \{\tilde{x}_i\}_{i=1}^M, \{d_{i,j}\}_{i,j=1}^M).$$

The result $\pi^* \in \Pi$ is the route used for the repair crew. In the simultaneous formulation, the model f^*_λ must not have a high graph traversal cost, and must yield probability estimates that agree with past observations. The constant C_1 is a tradeoff parameter between the accuracy of the predictive model and the cost to traverse the graph. C_1 needs to be scaled relative to the training error, m, M, and distances $d_{i,j}$. If C_1 is very small or if the first term is very large compared to the second, the algorithm essentially becomes sequential; first the training error is minimized without knowledge of the possible repair routes, and the graph traversal route would be the same as if it were determined after the model f^*_λ. In some cases, this may be appropriate, for instance if the number of training examples is extremely large, then there may be little flexibility in the choice of model f^*_λ.

In what follows, we define the TrainingError and GraphTraversalCost objectives.

2.1 Training Error Term

The training error term includes a sum of losses over the training examples:

$$\sum_{i=1}^m l_f(x_i, y_i),$$

where the loss function l_f can be any monotonically decreasing differentiable function bounded below by zero. We choose the logistic loss: $l_f(x, y) := \ln\left(1 + e^{-y f_\lambda(x)}\right)$ so that the probability of failure $P(y = 1|x)$, is estimated as in logistic regression by:

$$\hat{P}(y = 1|x) = p(x) := \frac{1}{1 + e^{-f_\lambda(x)}}. \tag{2}$$

We are thus assuming that the log-odds ratio of the class posterior values $P(y = \cdot|x)$ can be represented by an affine/linear function of the features x. The training error corresponds

to the negative log likelihood:

$$-\text{log likelihood } = \sum_{i=1}^{m} -\ln\left[p(x_i)^{(1+y_i)/2}(1-p(x_i))^{(1-y_i)/2}\right] = \sum_{i=1}^{m} \ln\left(1+e^{-y_i f_\lambda(x_i)}\right).$$

We include an ℓ_2 penalty over the parameters λ. The regularized training loss is now:

$$\text{TrainingError}(f_\lambda, \{x_i, y_i\}_{i=1}^m) := \sum_{i=1}^{m} \ln\left(1+e^{-y_i f_\lambda(x_i)}\right) + C_2||\lambda||_2^2 \tag{3}$$

where C_2 is the corresponding regularization coefficient. Another possible loss function is the exponential loss $e^{-y_i f_\lambda(x_i)}$, used in boosting (Schapire and Freund, 2011), which also corresponds to a probability model, though we will not use it here.

2.2 Two Options for the Graph Traversal Cost

The graph traversal cost can be defined to match the application. We present two options. In Cost 1, for each node there is a cost for failure event $\{y = 1\}$ in every time step prior to a visit by the repair crew. There can be repeated failures, where each failure has the same cost. In Cost 2, there is a cost for the first event on a node prior to its visit by the repair crew. Cost 2 assumes that the repair crew wants to reach the node before its first failure, and that there are no additional failures after the first one. Both Cost 1 and Cost 2 can be appropriate for various power grid applications. Cost 2 is appropriate for the delivery truck application, where perishable items can fail (once an item has spoiled, it cannot spoil again). Both graph traversal costs use the predictions on the nodes, $f_\lambda(\tilde{x}_i)$.

We can provide a very natural interpretation in terms of a simple stochastic process for both these costs. Consider that there is a continuous time stochastic process at each node $\tilde{x}_i$, which when discretized by time steps of appropriate duration, can be approximated by a Bernoulli process with parameter $p(\tilde{x}_i)$. We will see later on that for Cost 2, the random variables at each time step are required to be independent whereas for Cost 1 no such restriction is necessary (because of linearity of expectations). Such a stochastic process perspective is useful when one designs their own, possibly more complex, cost models.

We assume for convenience and without loss of generality that after the repair crew visits all the nodes, it returns to the location of the starting node, which is fixed beforehand to be node 1 ($\pi(1) = 1$). This assumption simplifies the exposition of the paper, but in doing so excludes the scenarios where one might not be interested in returning to the starting node or when one wants to start from a separate depot. Either of these cases would require a slight change in the cost model, as discussed for instance by Ezzine et al. (2010), and would not change the computational complexity of finding a solution. Another setting which one might be interested in is of finding an optimal tour route without specifying a starting point. In this case, problem formulation with a fixed starting node can be readily used. In particular, we can obtain a solution by choosing each node in turn to be the starting node, solving with a fixed starting node formulation M times, and picking the best of the M solutions. Because such extensions can be performed relatively easily, we are assuming a fixed known starting node in our formulations.

Let a route be represented by $\pi : \{1, ..., M\} \mapsto \{1, ..., M\}$, this means that $\pi(i)$ is the i^{th} node to be visited. For example, let $M = 4, \pi = [2, 3, 4, 1]$. This means, $\pi(1) = 2$, node

2 is the first node to be visited, $\pi(2) = 3$, node 3 is the second node on the route, and so on. The *latency* of a node $\pi(i)$ with respect to route π is the time (or equivalently distance) at which node $\pi(i)$ is visited. It is the sum of distances traversed before position i on the route:

$$L_\pi(\pi(i)) := \text{time at which node } \pi(i) \text{ is visited} = \begin{cases} \sum_{k=1}^{M} d_{\pi(k)\pi(k+1)} \mathbf{1}_{[k<i]} & i = 2, ..., M \\ \sum_{k=1}^{M} d_{\pi(k)\pi(k+1)} & i = 1. \end{cases} \tag{4}$$

The assumption that the final node is the first node means that $d_{\pi(M)\pi(M+1)} = d_{\pi(M)\pi(1)}$. The starting node $\pi(1)$ thus has a latency $L_\pi(\pi(1))$ which is the total length of the route starting at node $\pi(1)$ and ending at node $\pi(1)$ after visiting all other nodes.

Cost 1: Cost is Proportional to Expected Number of Failures Before the Visit

Up to the time that node $\pi(i)$ is visited by the repair crew, there is a probability $p(\tilde{x}_{\pi(i)})$ that a failure will occur within each unit time interval. Equivalently, within each unit time interval, failures are determined by a Bernoulli random variable with parameter $p(\tilde{x}_{\pi(i)})$. Thus, in a time interval of length $L_\pi(\pi(i))$ units, the number of node failures follows the Binomial distribution $\text{Bin}\left(L_\pi(\pi(i)), p(\tilde{x}_{\pi(i)})\right)$. For each node, we will associate a cost proportional to the expected number of failures before the repair crew's visit, as follows:

$$\begin{aligned} \text{Cost of node } \pi(i) &\propto E(\text{number failures in } L_\pi(\pi(i)) \text{ time units}) \\ &= \text{mean of } \text{Bin}(L_\pi(\pi(i)), p(\tilde{x}_{\pi(i)})) = p(\tilde{x}_{\pi(i)}) L_\pi(\pi(i)). \end{aligned} \tag{5}$$

Using this cost, if the failure probability for node $\pi(i)$, namely $p(\tilde{x}_{\pi(i)})$, is small, we can afford to visit it later on during our graph tour when the latency $L_\pi(\pi(i))$ is larger. If $p(\tilde{x}_{\pi(i)})$ is large, we should visit node $\pi(i)$ earlier to keep our overall graph traversal cost low.

The total cost of route π is:

$$\text{GraphTraversalCost}(\pi, f_\lambda, \{\tilde{x}_i\}_{i=1}^M, \{d_{i,j}\}_{i,j=1}^M) = \sum_{i=1}^{M} p(\tilde{x}_{\pi(i)}) L_\pi(\pi(i)).$$

Substituting the definition of $L_\pi(\pi(i))$ from (4):

$$\begin{aligned} &\text{GraphTraversalCost}(\pi, f_\lambda, \{\tilde{x}_i\}_{i=1}^M, \{d_{i,j}\}_{i,j=1}^M) = \\ &\sum_{i=2}^{M} p(\tilde{x}_{\pi(i)}) \sum_{k=1}^{M} d_{\pi(k)\pi(k+1)} \mathbf{1}_{[k<i]} + p(\tilde{x}_{\pi(1)}) \sum_{k=1}^{M} d_{\pi(k)\pi(k+1)}, \end{aligned} \tag{6}$$

where $p(\tilde{x}_{\pi(i)})$ is given in (2). This will be Cost 1.

There are ways to make Cost 1 more general. The individual node cost in (5) assumes that the node failure probability $p(\tilde{x}_{\pi(i)})$ becomes zero after the repair crew's visit, so that for the remainder of the route, the cost incurred at this node is $\propto 0 \times (L_\pi(\pi(1)) - L_\pi(\pi(i))$. We could relax this by assuming $p(\tilde{x}_{\pi(i)})$ does not vanish after the repair crew's visit and

adding an additional cost for the expected failures in this period. That is, if β is a constant of proportionality for the cost after visiting node $\pi(i)$, then the cost would become:

$$\text{Cost of node } \pi(i) = \beta\left[L_\pi(\pi(1)) - L_\pi(\pi(i))\right] p(\tilde{x}_{\pi(i)}) + L_\pi(\pi(i)) p(\tilde{x}_{\pi(i)}).$$

If $\beta = 1$, then the repair crew does not have any effect and cost of each node is independent of its expected number of failures before the repair crew's visit. Typically, we expect that the repair crew will repair the node so that it will not fail, and the second term above is much larger than the first. Taking the constant of proportionality as $\beta = 0$, we return to the individual costs given by (5).

Note that since the cost is a sum of M terms, it is invariant to ordering or indexing (caused by π). Thus we can rewrite the cost as

$$\text{GraphTraversalCost}(\pi, f_\lambda, \{\tilde{x}_i\}_{i=1}^M, \{d_{i,j}\}_{i,j=1}^M) = \sum_{i=1}^M p(\tilde{x}_i) L_\pi(i),$$

where $p(\tilde{x}_{\pi(i)})$ is given in (2).

Cost 2: Cost is Proportional to Probability that First Failure is Before the Visit

This cost reflects the penalty for not visiting a node before the first failure occurs there. This model is governed by the geometric distribution: the probability that the first failure for node $\pi(i)$ occurs at time $L_\pi(\pi(i))$ is $p(\tilde{x}_{\pi(i)})(1 - p(\tilde{x}_{\pi(i)}))^{L_\pi(\pi(i))-1}$, and:

$$P\Big(\text{first failure occurs before time } L_\pi(\pi(i))\Big) \;=\; 1 - (1 - p(\tilde{x}_{\pi(i)}))^{L_\pi(\pi(i))}. \tag{7}$$

The cost of visiting node $\pi(i)$ will be proportional to this quantity. Substituting the expression (2) for $p(\tilde{x}_{\pi(i)})$:

$$\begin{aligned} \text{Cost of node } \pi(i) \;&\propto\; \left(1 - \left(1 - \frac{1}{1 + e^{-f_\lambda(\tilde{x}_{\pi(i)})}}\right)^{L_\pi(\pi(i))}\right) \\ &=\; \left(1 - \left(1 + e^{f_\lambda(\tilde{x}_{\pi(i)})}\right)^{-L_\pi(\pi(i))}\right). \end{aligned} \tag{8}$$

Similarly to Cost 1, $L_\pi(\pi(i))$ influences the cost at each node. If we visit a node early in the route, then the cost incurred is small because the node is less likely to fail before we reach it. Similarly, if we schedule a visit later on in the tour, the cost is higher because the node has a higher chance of failing prior to the repair crew's visit. We generally want to visit nodes with higher probability of failures early and schedule the less vulnerable nodes later. The total route cost is thus:

$$\text{GraphTraversalCost}(\pi, f_\lambda, \{\tilde{x}_i\}_{i=1}^M, \{d_{i,j}\}_{i,j=1}^M) = \sum_{i=1}^M \left(1 - \left(1 + e^{f_\lambda(\tilde{x}_{\pi(i)})}\right)^{-L_\pi(\pi(i))}\right). \tag{9}$$

This cost is not directly related to a TRP cost in its present form. That is, when the failure probabilities of the nodes are all the same, the total cost is not linear in the latencies, as is the case for Cost 1. Loosely inspired by the route cost above, we will derive a cost that is similar to a weighted version of the TRP in Section 3.2, choosing it to be of the form:

$$\text{Cost of node } \pi(i) \quad \propto \quad L_\pi(\pi(i)) \log\left(1 + e^{f_\lambda(\tilde{x}_{\pi(i)})}\right), \tag{10}$$

as an alternative to (8).

There is a slightly more general version of this formulation (as there was for Cost 1), which is to take the cost for each node to be a function of two quantities: the probability of failure before the visit, and the probability of failure after the visit. Let us redefine β to be a constant of proportionality for the cost of visiting before the failure event. From the geometric distribution, $P(\text{first failure occurs after time } L_\pi(\pi(i)) = (1-p(\tilde{x}_{\pi(i)}))^{L_\pi(\pi(i))}$, and the cost of visiting node $\pi(i)$ becomes:

$$\text{Cost of node } \pi(i) \quad \propto \quad P(\text{failure before } L_\pi(\pi(i))) + \beta P\left(\text{failure after } L_\pi(\pi(i))\right). \tag{11}$$

If $\beta = 1$, then the sum above is 1 for all nodes regardless of node failures or latencies. More realistically, the cost of visiting the node after the failure is more than the cost of visiting proactively, $\beta \ll 1$ leading to (8).

We could again have written the summation to hide the dependence on π:

$$\text{GraphTraversalCost}(\pi, f_\lambda, \{\tilde{x}_i\}_{i=1}^M, \{d_{i,j}\}_{i,j=1}^M) = \sum_{i=1}^M \left(1 - \left(1 + e^{f_\lambda(\tilde{x}_i)}\right)^{-L_\pi(i)}\right).$$

Now that the major steps for both formulations have been defined, we will discuss methods for optimizing the objectives.

3. Optimization

We start by formulating mixed-integer linear programs (MILP's) for the graph traversal cost subproblem.

3.1 Mixed-integer optimization for Cost 1

For either the sequential or simultaneous formulations, we need the solution of the subproblem:

$$\begin{aligned} \pi^* \quad &\in \quad \text{argmin}_{\pi\in\Pi}\text{GraphTraversalCost}(\pi, f_\lambda^*, \{\tilde{x}_i\}_{i=1}^M, \{d_{i,j}\}_{i,j=1}^M), \\ &= \quad \text{argmin}_{\pi\in\Pi} \sum_{i=2}^M p(\tilde{x}_{\pi(i)}) \sum_{k=1}^M d_{\pi(k)\pi(k+1)} \mathbf{1}_{[k<i]} + p(\tilde{x}_{\pi(1)}) \sum_{k=1}^M d_{\pi(k)\pi(k+1)}. \end{aligned} \tag{12}$$

Let us compare this to the standard traveling repairman problem (TRP) problem (see Blum et al., 1994):

$$\pi^* \in \text{argmin}_{\pi\in\Pi} \sum_{k=1}^M d_{\pi(k)\pi(k+1)}(M+1-k). \tag{13}$$

The standard TRP objective (13) is a special case of weighted TRP (12) when $\forall i = 1, ..., M,\ p(\tilde{x}_i) = p$:

$$\begin{aligned}
&\sum_{i=2}^{M} p(\tilde{x}_{\pi(i)}) \sum_{k=1}^{M} d_{\pi(k)\pi(k+1)} \mathbf{1}_{[k<i]} + p(\tilde{x}_{\pi(1)}) \sum_{k=1}^{M} d_{\pi(k)\pi(k+1)} \\
&= p \sum_{i=2}^{M} \sum_{k=1}^{M} d_{\pi(k)\pi(k+1)} \mathbf{1}_{[k<i]} + p \sum_{k=1}^{M} d_{\pi(k)\pi(k+1)} \\
&= p \sum_{i=2}^{M} \sum_{k=1}^{M} d_{\pi(k)\pi(k+1)} \mathbf{1}_{[k<i]} + p \sum_{k=1}^{M} d_{\pi(k)\pi(k+1)} \mathbf{1}_{[k<M+1]} \\
&= p \sum_{k=1}^{M} d_{\pi(k)\pi(k+1)} \sum_{i=2}^{M+1} \mathbf{1}_{[k<i]} \\
&= p \sum_{k=1}^{M} d_{\pi(k)\pi(k+1)} (M + 1 - k).
\end{aligned}$$

The TRP is different from the traveling salesman problem (TSP); the goal of the traveling salesman problem is to minimize the total traversal time (in this case, this is the same as the distance traveled) needed to visit all nodes once, whereas the goal of the traveling repairman problem is to minimize the sum of the waiting times to visit each node. Both the TSP and the TRP are known to be NP-complete in the general case (Blum et al., 1994). Intuitively, a TRP route cost objective captures the total waiting cost of a service system from the customer's (the node's) point of view. For example, consider a truck carrying prioritized items to be delivered to customers. At each customer's stop, that customer's item is removed from the truck. The goal of the TRP is to minimize the total waiting time of these customers.

We need to extend the standard TRP to include "unequal flow values" that will accommodate the more general problem (12). We use as a starting point the work of Fischetti et al. (1993) who give an integer programming formulation of the standard TRP. (Note that there are usually many ways that an integer program can be constructed, see Méndez-Díaz et al., 2008). We will suitably introduce pre-defined weights $\{\bar{p}(\tilde{x}_i)\}_i$ to arrive at (12) as the objective. For Cost 1, we will take $\bar{p}(\tilde{x}_i) := p(\tilde{x}_i) = 1/\left(1 + \exp\left(-f_\lambda(\tilde{x}_i)\right)\right)$, though we leave the weights in the formulation more generally as $\bar{p}(\tilde{x}_i)$. In order to interpret the formulation below, consider the sum of the probabilities $\sum_{i=1}^{M} \bar{p}(\tilde{x}_i)$ as the total "flow" through a route. At the beginning of the tour, the repair crew has flow $\sum_{i=1}^{M} \bar{p}(\tilde{x}_i)$. Along the tour, flow of the amount $\bar{p}(\tilde{x}_i)$ is dropped when the repair crew visits node i at latency $L_\pi(\pi(i))$. In this way, the amount of flow during the tour is the sum of the probabilities $\bar{p}(\tilde{x}_i)$ for nodes that the repair crew has not yet visited.

We deviate from the π notation to represent routes in the mixed-integer program. Instead, we introduce two sets of variables $\{z_{i,j}\}_{i,j}$ and $\{y_{i,j}\}_{i,j}$ which can together represent a route. Let the set of edges of graph G be denoted by E. Let $z_{i,j}$ represent the flow on edge $(i,j) \in E$ and let a binary variable $y_{i,j}$ represent whether there exists a flow on edge $(i,j) \in E$. (There will only be a flow along the route, and there will not be a flow along

edges that are not in the route.) The mixed-integer program is as follows:

$$\min_{z,y} \sum_{i=1}^{M} \sum_{j=1}^{M} d_{i,j} z_{i,j} \quad \text{s.t.} \tag{14}$$

$$\text{No flow from node } i \text{ to itself: } \; z_{i,i} = 0 \;\; \forall i = 1, ..., M \tag{15}$$

$$\text{No edge from node } i \text{ to itself: } \; y_{i,i} = 0 \;\; \forall i = 1, ..., M \tag{16}$$

$$\text{Exactly one edge into each node: } \; \sum_{i=1}^{M} y_{i,j} = 1 \;\; \forall j = 1, ..., M \tag{17}$$

$$\text{Exactly one edge out from each node: } \; \sum_{j=1}^{M} y_{i,j} = 1 \;\; \forall i = 1, ..., M \tag{18}$$

$$\text{Flow coming back to the initial point at the end of the loop is } \bar{p}(\tilde{x}_1): \; \sum_{i=1}^{M} z_{i,1} = \bar{p}(\tilde{x}_1) \tag{19}$$

Change of flow after crossing node k is either $\bar{p}(\tilde{x}_k)$ or it is $\bar{p}(\tilde{x}_1)$ minus the sum of $\bar{p}$'s:

$$\sum_{i=1}^{M} z_{i,k} - \sum_{j=1}^{M} z_{k,j} = \begin{cases} \bar{p}(\tilde{x}_1) - \sum_{i=1}^{M} \bar{p}(\tilde{x}_i) & k = 1 \\ \bar{p}(\tilde{x}_k) & k = 2, ..., M \end{cases} \tag{20}$$

$$\text{Connects flows } z \text{ to indicators of edge } y: \; z_{i,j} \leq r_{i,j} y_{i,j} \tag{21}$$

$$\text{where } r_{i,j} = \begin{cases} \bar{p}(\tilde{x}_1) & j = 1 \\ \sum_{i=1}^{M} \bar{p}(\tilde{x}_i) & i = 1 \\ \sum_{i=2}^{M} \bar{p}(\tilde{x}_i) & \text{otherwise} \end{cases}$$

Constraints (15) and (16) restrict self-loops from forming. Constraints (17) and (18) impose that every node should have exactly one edge coming in and one going out. Constraint (19) represents the flow on the last edge coming back to the starting node. Constraint (20) quantifies the flow change after traversing a node k. Constraint (21) represents an upper bound on $z_{i,j}$ relating it to the corresponding binary variable $y_{i,j}$.

3.2 Mixed integer optimization for Cost 2

Here we reason about the choice for changing Cost 2 in (8) to resemble (10). Starting with the sum (9) over costs (8),

$$\min_{\pi} \sum_{i=1}^{M} \left(1 - \left(1 + e^{f_\lambda(\tilde{x}_{\pi(i)})} \right)^{-L_\pi(\pi(i))} \right),$$

we apply the log function to the cost of each node (8) to get a new cost

$$\left(1 - \log \left(1 + e^{f_\lambda(\tilde{x}_{\pi(i)})} \right)^{-L_\pi(\pi(i))} \right),$$

and the minimization becomes instead:

$$\min_{\pi} \sum_{i=1}^{M} \left(1 - \log \left(1 + e^{f_\lambda(\tilde{x}_{\pi(i)})} \right)^{-L_\pi(\pi(i))} \right)$$

$$
\begin{aligned}
&= -\max_{\pi}\left(\sum_{i=1}^{M}\log\left(1+e^{f_\lambda(\tilde{x}_{\pi(i)})}\right)^{-L_\pi(\pi(i))}-M\right)\\
&= -\max_{\pi}\left(\sum_{i=1}^{M}(-L_\pi(\pi(i))\log\left(1+e^{f_\lambda(\tilde{x}_{\pi(i)})}\right)-M\right)\\
&= \min_{\pi}\sum_{i=1}^{M}L_\pi(\pi(i))\log\left(1+e^{f_\lambda(\tilde{x}_{\pi(i)})}\right)+M,
\end{aligned}
$$

which is the sum over nodes of the expression (10). This graph traversal cost term is now a weighted sum of latencies ($+M$) where the weights are of the form $\log\left(1+e^{f_\lambda(\tilde{x}_{\pi(i)})}\right)$. We can thus reuse the mixed integer program (14)-(21) where the weights are defined as $\bar{p}(\tilde{x}_i) := \log\left(1+e^{\lambda\cdot\tilde{x}_i}\right)$.

The sequential formulation has now been completely defined for both Cost 1 and Cost 2.

3.3 Solvers for the weighted TRP subproblem

A generic MILP solver like CPLEX[1] or Gurobi[2] can produce an exact solution using branch-and-bound or other related exact methods. In our experiments we use Gurobi. The weighted TRP problem is NP-hard (can be shown by a reduction of the hamiltonian cycle problem) and hence most likely not solvable by polynomial-time algorithms. The standard (unweighted – all weights equal) TRP can be encoded by different mixed-integer programming formulations (see Fischetti et al., 1993; Eijl van, 1995; Méndez-Díaz et al., 2008) each with different performance guarantees (e.g., solving 15-60 nodes), which could be adapted for our purpose. There are also techniques for producing constant factor approximate solutions to the unweighted TRP, which could run faster than the MILP solvers for large problem instances. If the weights are integers, we can adapt these faster techniques for the standard problem to the weighed TRP problem by replicating the respective nodes by w_i times where w_i is equal to the weight of node i. If the weights are rational (as is the case in (22) and (23)), a rounding and discretization trick is one way to map back to the standard solution techniques. More on this topic is discussed in Section 6.

3.4 Mixed-integer nonlinear programs (MINLPs)

For the simultaneous formulation, the inputs to the program are training data $\{x_i, y_i\}_{i=1}^m$, unlabeled nodes $\{\tilde{x}_i\}_{i=1}^M$ the distances between them $\{d_{i,j}\}_{i,j=1}^M$ and constants C_1 and C_2. The full objective using Cost 1 is:

$$
\min_{\lambda,\{z_{i,j},y_{i,j}\}}\left(\sum_{i=1}^{m}\ln\left(1+e^{-y_i f_\lambda(x_i)}\right)+C_2||\lambda||_2^2+C_1\sum_{i=1}^{M}\sum_{j=1}^{M}d_{i,j}z_{i,j}\right)\quad\text{s.t.}
$$

$$
\text{constraints (15) to (21) hold, where } \bar{p}(\tilde{x}_i)=\frac{1}{1+e^{-\lambda\cdot\tilde{x}_i}},
$$

1. IBM ILOG CPLEX Optimization Studio v12.2.0.2 2010
2. Gurobi Optimizer v3.0, Gurobi Optimization, Inc. 2010

or equivalently,

$$\min_{\lambda} \left(\sum_{i=1}^{m} \ln\left(1 + e^{-y_i f_\lambda(x_i)}\right) + C_2||\lambda||_2^2 + C_1 \min_{\{z_{i,j}, y_{i,j}\}} \sum_{i=1}^{M}\sum_{j=1}^{M} d_{i,j} z_{i,j} \right) \quad \text{s.t.} \tag{22}$$

$$\text{constraints (15) to (21) hold, where } \bar{p}(\tilde{x}_i) = \frac{1}{1 + e^{-\lambda \cdot \tilde{x}_i}}.$$

The full objective using the modified version of Cost 2 is:

$$\min_{\lambda} \left(\sum_{i=1}^{m} \ln\left(1 + e^{-y_i f_\lambda(x_i)}\right) + C_2||\lambda||_2^2 + C_1 \min_{\{z_{i,j}, y_{i,j}\}} \sum_{i=1}^{M}\sum_{j=1}^{M} d_{i,j} z_{i,j} \right) \quad \text{s.t.} \tag{23}$$

$$\text{constraints (15) to (21) hold, where } \bar{p}(\tilde{x}_i) = \log\left(1 + e^{\lambda \cdot \tilde{x}_i}\right).$$

If we have an algorithm for solving (22), then the same scheme can be used to solve (23). There are multiple ways of solving (or approximately solving) a mixed integer nonlinear optimization problem of the form (22) or (23). We consider three methods here, described next. The first method is to directly use a generic mixed integer non-linear programming (MINLP) solver. The second and third methods (called Nelder-Mead and Alternating Minimization, denoted NM and AM respectively) are iterative schemes over the λ parameter space. At every iteration of these algorithms, we will need to evaluate the objective function. This evaluation involves solving the weighted TRP subproblem, as discussed in the previous subsections.

Method 1: MINLP Solver

For our experiments we directly use a MINLP solver called Bonmin (Bonami et al., 2008). These types of solvers typically use general MILP solving techniques like branch and bound or dynamic programming interleaved with continuous optimization. Since the general MILP solving techniques, as discussed, can take exponential time when applied directly to our formulations, the MINLP solvers which use them can in turn, be inefficient if the graph is moderate to large in size. However, when the graph is small, for instance when we want to schedule a tour over a small time period with only a few nodes, the MINLP solver can directly compute a solution to the problems (22) or (23) in manageable time.

Method 2: Nelder-Mead in λ-space (NM)

The Nelder-Mead minimization algorithm requires only function evaluations (Nelder and Mead, 1965). The ML&TRP can be viewed as a minimization in the space of all λ vectors; since we have solvers for the weighted TRP subproblem, we are able to evaluate the ML&TRP objective for a given value of λ. In our experiments we use the MILP solver (Gurobi) for the subproblem. Note that the ML&TRP objective can have non-differentiable kinks arising from discontinuities in the graph traversal cost term; a method which relies on the gradient or the Hessian of the objective function might get stuck. By using only function values, NM may be able to bypass this type of situation. The generic Nelder-Mead scheme can have disadvantages Rios (2009) with respect to performance. If so, other schemes like

Multilevel Coordinated Search (MCS) Huyer and Neumaier (1999) can be used in place of Nelder-Mead. Note that since the objective is nonlinear, all solutions obtained by NM are only locally optimal.

Method 3: Alternating minimization in λ-π space (AM)

Define Obj as follows:

$$\text{Obj}(\lambda, \pi) = \text{TrainingError}(f_\lambda, \{x_i, y_i\}_{i=1}^m) + C_1 \text{GraphTraversalCost}\left(\pi, f_\lambda, \{\tilde{x}_i\}_{i=1}^M, \{d_{i,j}\}_{i,j=1}^M\right). \tag{24}$$

We propose a heuristic minimization algorithm, where starting from an initial vector λ_0, Obj is minimized alternately with respect to λ and then with respect to π, as shown in Algorithm 1. The second step, solving for π, is the same as solving the TRP subproblem, and we again use the MILP solver for this.

Algorithm 1 AM: Alternating minimization algorithm

Inputs: $\{x_i, y_i\}_1^m, \{\tilde{x}_i\}_1^M, \{d_{ij}\}_{ij}, C_1, C_2, T$ and initial vector λ_0.
for t=1:T **do**
  Compute $\pi_t \in \text{argmin}_{\pi \in \Pi} \text{Obj}(\lambda_{t-1}, \pi)$.
  Compute $\lambda_t \in \text{argmin}_{\lambda \in \mathbb{R}^d} \text{Obj}(\lambda, \pi_t)$.
end for
Output: π_T.

Conditions for convergence and correctness for such iterative schemes is given by Csiszár and Tusnády (1984); only locally optimal solutions can be found using this method.

4. Experiments

One of the major goals of this work is to be able to produce low cost solutions that are still high quality; these models can explain the variance in the training data, while promoting the prior belief that the cost will be low for carrying out the model's recommendations. The sequential formulation will not necessarily be able to accomplish this: the best possible minimizer of the training error will not necessarily yield a low cost solution. This point is made through some illustrations that follow next. We then evaluate the two formulations with respect to both accuracy on a test set and cost of the route, for a set of features derived from data from New York City's secondary electrical distribution network.

4.1 Illustrations

Our first illustration shows how a small change in the probabilities can give a completely different route and change the traversal cost. The graph G given by $\{d_{i,j}\}_{i,j}$ is shown in Figure 1. The number of unlabeled nodes is $M = 4$, $\tilde{x}_1, \ldots, \tilde{x}_4 \in \mathbb{R}^2$, shown in Figure 2. The training features are also in the plane, $x_i \in \mathbb{R}^2$, and are represented by two gray circles in Figure 2 (for instance, the distributions could be Normal with diameters representing their corresponding variance). Note that it is important not to confuse the feature space of x_i's with the space that the graph $\{d_{i,j}\}_{i,j}$ is embedded in, these are different spaces, and the ML&TRP graph need not even have a physical distance interpretation.

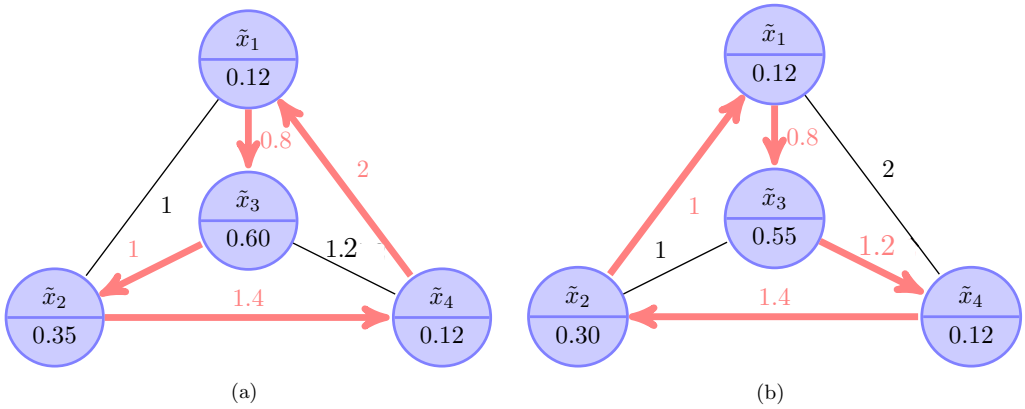


Figure 1: Physical space for the four node illustration. The weights on the edges represent the distance $d_{i,j}$. The optimal route as determined by the sequential formulation is highlighted in (a). (b) shows a route determined by the simultaneous formulation.

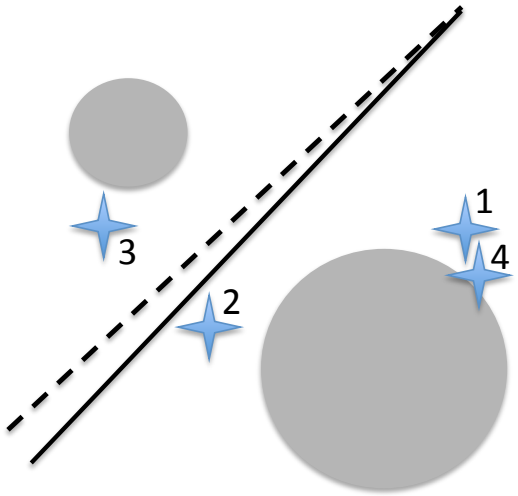


Figure 2: Feature space for the four node illustration.

The sequential formulation produces a function whose 0.5-probability level set is displayed (in feature space) as a black line in Figure 2. The route corresponding to that solution is given in Figure 1(a), which is $\pi^* = 1-3-2-4-1$. Consider instead what might happen if we used the simultaneous formulation. If we were to move the 0.5-probability level set slightly, for instance to the dashed line in Figure 2, the probability estimates on the finite training set would change only slightly, but the route would change entirely. The new route would be $\pi^*_{new} = 1-3-4-2-1$, and it would yield a lower value of Cost 1 (decrease by $\sim 16.4\%$). In both cases, the probability estimators may have very similar validation performance, so a solution from the simultaneous formulation might be preferred.

For our second illustration, we consider a new set of experiments with a similar setup as before but now with the number of nodes on the graph equal to 6. The training set was chosen uniformly at random from a distribution that is uniform over two triangles pointing end to end. Again the training data is finite, so that the level set can be moved, yielding almost the same probability estimates on the training set but accommodating lower costing routes. Figure 3 shows the training instances and unlabeled instances in feature space along with two level sets. The first one, colored black drawn at probability estimate 0.5 is learned from (ℓ_2-regularized) logistic regression. The second level set, colored red and also drawn at probability estimate 0.5 is learned from the new simultaneous formulation. The new level set was obtained from the simultaneous formulation with graph cost modeled according to Cost 1 (with an appropriately chosen coefficient C_1). Node 6 lies in a low density region of feature space, so its probability cannot be well estimated. In the sequential formulation, node 6 which was assigned $p(\tilde{x}_6) = 0.5$. The optimal route thus obtained by solving the weighted TRP problem in the second step is $1-2-3-6-4-5-1$ shown in Figure 4. In the simultaneous formulation, node 6 has been assigned a new probability value $p(\tilde{x}_6) = 0.29$. This big change is possible because its probability estimate can vary quite a lot without changing the probability estimates of others. This changes the route to $1-2-3-4-5-6-1$ as shown in Figure 5. Here, we see that node 6 is also physically far from all other nodes. If it has a high enough probability estimate compared to nodes 4 and 5 (blue triangles in the lower left half of Figure 3), then a route that visits node 6 before visiting nodes 4 and 5 would be favored; this is what happens in the sequential formulation. In the simultaneous formulation, we chose C_1 large enough so that the tour route visits 4 and 5 before 6. This results in $\sim 9\%$ decrease in the route cost (Cost 1).

Using the data from the second illustration, the (ℓ_2-regularized) training error is plotted in Figure 6(a). The axes are the first two coordinates of the λ parameter vector. The optimal graph traversal cost (Cost 1) was computed for each value of λ and is plotted in Figure 6(b); for each point, a weighted TRP subproblem was solved. The simultaneous ML&TRP objective is the sum of the values in Figures 6(a) and 6(b), and the constant C_1 controls how these surfaces are added together. If the training error term in Figure 6(a) is somewhat flat near the minimizer of the ML&TRP objective and the graph traversal term in 6(b) is not flat, the graph traversal term may be able to have a substantial effect on the solution.

4.2 ML&TRP on the NYC power grid

We now illustrate the performance of our method on a data set obtained from a collaborative effort with Con Edison, which is NYC's power utility company. More details about these data can be found in (Rudin et al., 2010). This dataset was developed in order to assist Con Edison with its maintenance and repair programs on the secondary electrical distribution network in NYC; specifically, it was designed for the purpose of predicting manhole fires and explosions. We chose to use all manholes from the Bronx ($\sim$23K manholes). Each manhole is represented by features that encode the number and type of electrical cables entering the manhole and the number and type of past events involving the manhole (e.g., if the manhole was the source of partial outages, full outages and/or underground burnouts). The training features encode events prior to 2008, and the training labels are 1 if the manhole

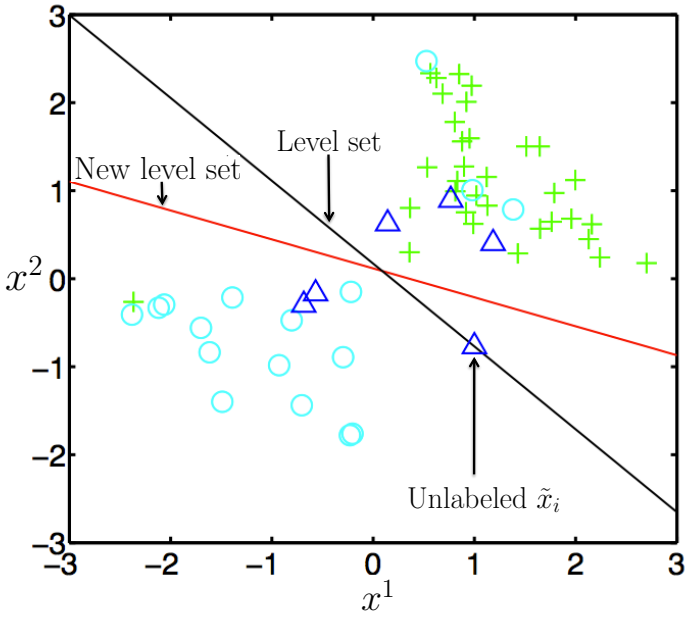


Figure 3: Plotting $\{x_i\}_{i=1}^m$ and $\{\tilde{x}_i\}_{i=1}^M$ in the feature space. Two level sets corresponding to 0.5 probability are also shown. The first (black) is obtained by minimizing the training error using the sequential method. The second (red) is the 0.5 probability level set obtained from the simultaneous formulation, and illustrates the effect of the graph traversal cost regularization term on the decision boundary.

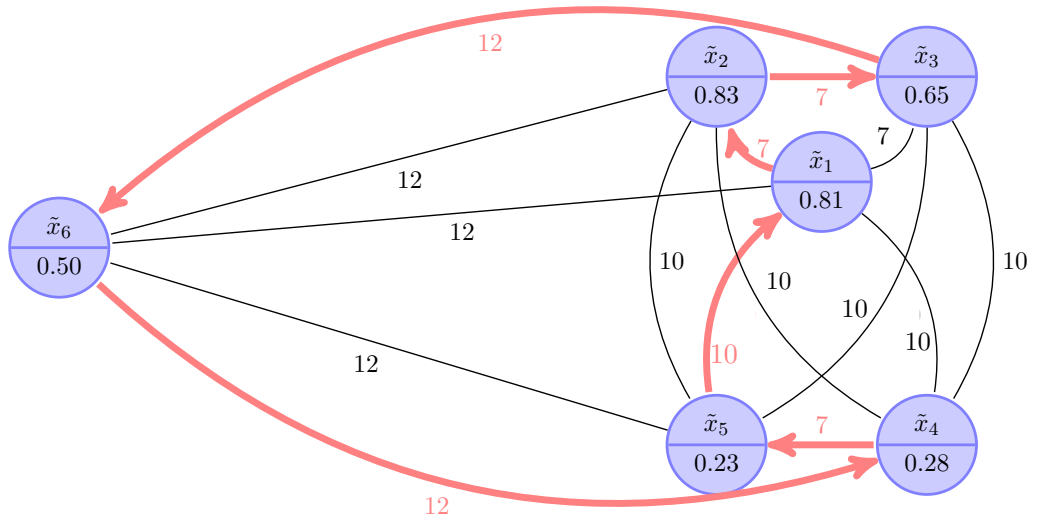


Figure 4: The weights on the edges represent the distance $d_{i,j}$. The optimal route $1-2-3-6-4-5-1$ as determined by the sequential formulation is highlighted. The route cost (Cost 1) is 4.7 units (scaled by $C_1 = 0.001$) and the training cost is 15.7 units. The values of $p(\tilde{x}_i)$ are shown in the node circles.

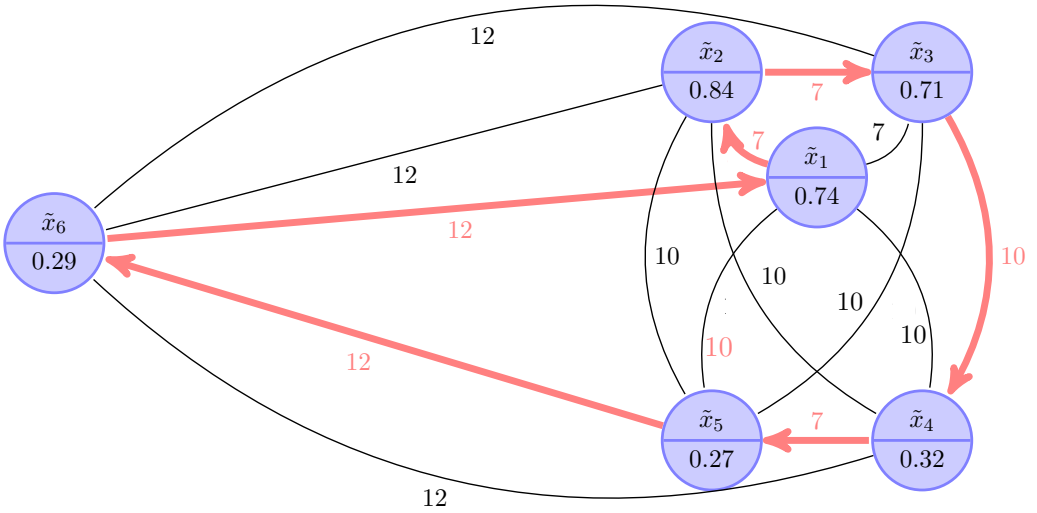
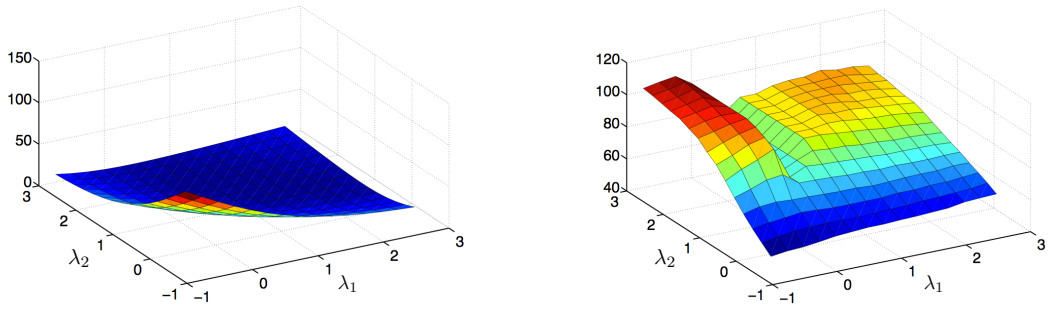


Figure 5: The optimal route $1-2-3-4-5-6-1$ as determined by the simultaneous formulation is highlighted. The route cost (Cost 1) is now 4.25 units (scaled by $C_1 = 0.001$) and the training cost is 16.2 units.



(a) (b)

Figure 6: The values of the two terms in the objective of the simultaneous ML&TRP formulation. (a) Training error as a function of $\{\lambda_1, \lambda_2\}$. The last coordinate, λ_3 is kept fixed. (b) Scaled optimal graph traversal cost (Cost 1 divided by 100) over a 2D grid of λ_1 and λ_2, again with λ_3 fixed.

was the source of a serious event (fire, explosion, smoke) during 2008. The nodes are 7 randomly chosen manholes, and the features for the nodes encode events prior to 2009. The prediction task is to predict events in 2009. The test set (for evaluating the performance of the predictive model) consists of features derived from the time period before 2009, and labels from 2009. Predicting manhole events is a difficult task for machine learning, because one cannot necessarily predict an event using the available data. The operational task was to design a route for a repair crew that is fixing the nodes. The choice of $M = 7$ nodes was only to speed up the computation time. Limitation on the number of nodes for which the TRP problem can be solved efficiently directly affects the number of nodes which we can pick for the unlabeled set. This bound is an order of magnitude more than the choice made here.

The distances between the nodes were obtained from Google Maps, by querying the driving distance between each pair of nodes. Note that we do not want 'flying' distance between two coordinates as this can be very different from the actual driving distance. Manhole failures are rare events. This means that we have many more negative labels than positive labels. Because of the large class imbalance, using a logistic model gives us probability estimates which are low overall, so the misclassification error is almost always the size of the whole positive class. To avoid this, we chose to evaluate the quality of the predictions from f_{λ^*} using the area under the ROC curve (AUC), for both training and test. The quality of the route is indicated by computing the optimal route cost at λ^*.

Figures 7 and 8 show how the AUC values change with respect to the coefficient C_1 of the graph traversal cost term in the objectives of Cost 1 and Cost 2. The algorithms used here are the Nelder-Mead method with the MILP solver for the subproblem, and the alternating minimization method (AM) again with the MILP solver. Having the graph traversal cost as a regularizer lowers predictor f_{λ^*}'s AUC values on the training data, as

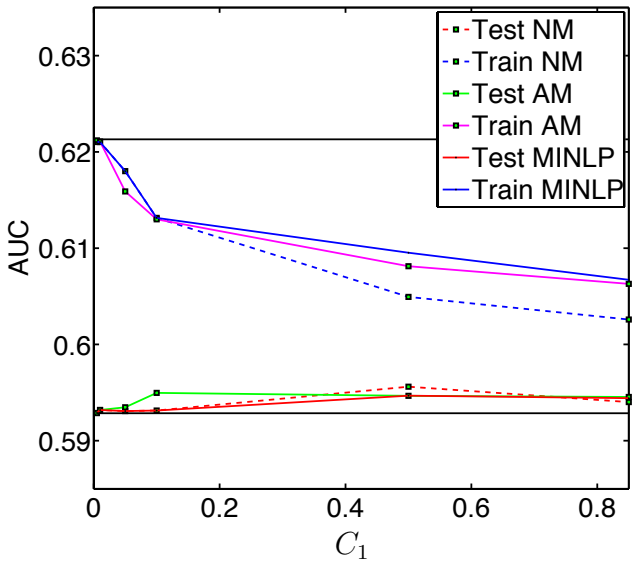


Figure 7: The AUC values corresponding to model parameters obtained from the simultaneous formulation using Cost 1 by NM-MILP and AM-MILP algorithms along with MINLP solver, plotted as a function of C_1. The AUC values on the training data decrease slightly and the same values for test data increase marginally. The two horizontal lines represent the training and test AUC values obtained by ℓ_2-penalized logistic regression and thus, are constant with respect to C_1.

expected. In this case, the performance on the test data is basically unchanged. We use a total of 4 features (that is, $x_i \in \mathbb{R}^4$). On the other hand, a related work on the same dataset (Rudin et al., 2011) uses more number of features and get about 10% increase in the AUC values. The increase in training error for the simultaneous formulation (using Cost 1) as a function of C_1 is shown also in Figure 9. The decrease in graph traversal cost as a function of C_1 is shown in Figure 10.

The naive route obtained by estimating probabilities using ℓ_2-penalized logistic regression, and then simply visiting nodes according to decreasing values of these probabilities is shown in Figure 11. Figure 12 shows the route provided by the sequential formulation. For the simultaneous method, there are changes in the route as the coefficient C_1 increases. When C_1 is low, the route is the same as obtained from the sequential method, in Figure 12. When the graph traversal term starts influencing the optimal solution of the objective (22) because of an increase in C_1, we get a new route, depicted in Figure 13.

We experimented with a large range of values for the regularization parameter C_1, with the goal of seeing a large range of possible results. We chose AUC as the evaluation metric, which is a measure of ranking quality; it is sensitive to the rank-ordering of the nodes in order of their probability to fail, and it is not as sensitive to changes in the values of these probabilities. This means that as the parameter C_1 increases, the estimated probability values will tend to decrease, and thus the graph traversal cost will decrease; however, it

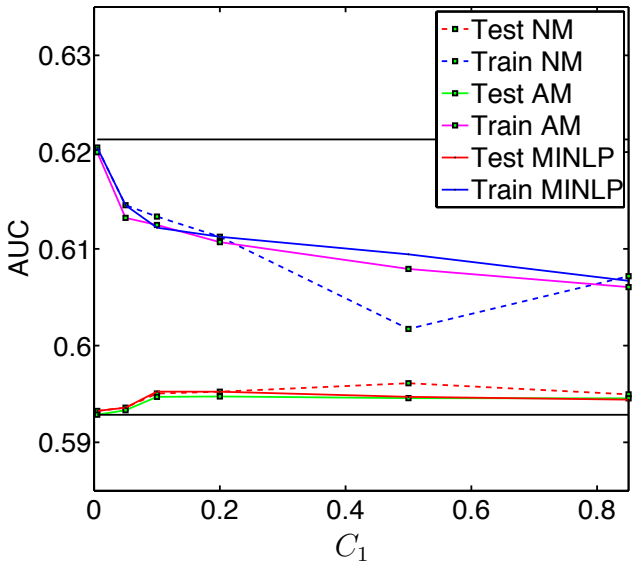


Figure 8: The AUC values obtained from the simultaneous formulation, using Cost 2, from the NM-MILP and AM-MILP algorithms along with the MINLP solver, plotted as a function of C_1. Again, the training data AUC values decrease and the test data AUC values remain nearly constant. The horizontal lines represent constant values of the AUC obtained by ℓ_2-penalized logistic regression.

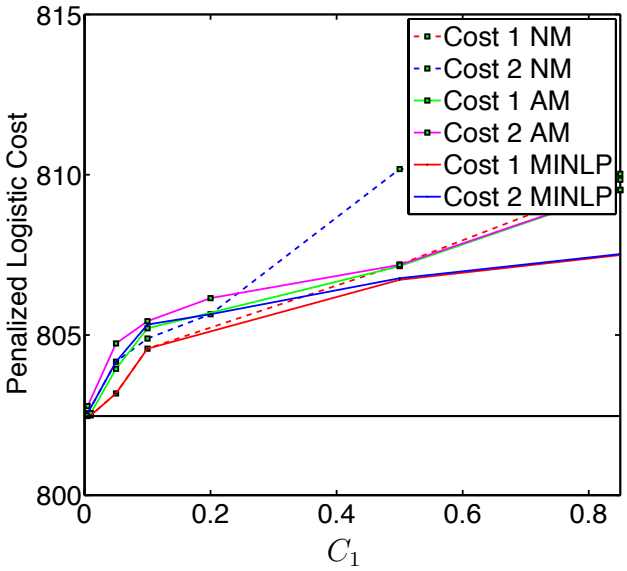


Figure 9: The ℓ_2-regularized logistic loss increases as a function of increasing C_1. The horizontal line represents the loss value from ℓ_2-penalized logistic regression with no regularization ($C_1 = 0$).

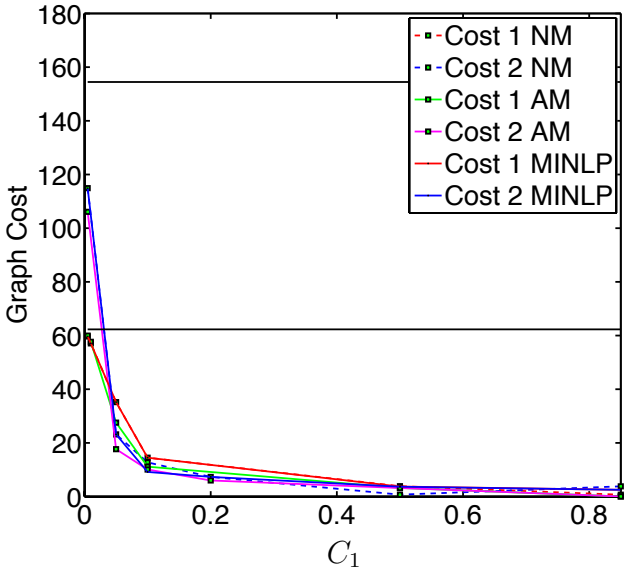


Figure 10: The graph traversal costs decrease as a function of the regularization parameter C_1. The horizontal lines in the figure represent the sequential formulation solutions; the lower horizontal line is Cost 1 of the solution obtained by ℓ_2-penalized logistic regression, and the upper line is Cost 2 of that solution.

Figure 11: A naive route: 1-5-4-3-2-6-7-1 obtained by sorting the probability estimates in decreasing order and visiting the corresponding nodes.

Figure 12: Sequential formulation route: 1-5-3-4-2-6-7-1. The simultaneous formulation also chooses this route when C_1 is small.

Figure 13: Route chosen by the simultaneous formulation when C_1 is larger: 1-6-7-5-3-4-2-1. Prediction performance is only slightly influenced by the route change, but the cost of the route (Cost 1) decreases a lot.

may be possible for this to happen without impacting the prediction quality as measured by the AUC, but this depends on the routes and it is not guaranteed. In our experiments, for both training and test we had a large sample ($\sim$23K examples). The test AUC values for the simultaneous method were all within 1% of the values obtained by the sequential method; this is true for both Cost 1 and Cost 2, for each of the AM, NM, and MINLP solvers. This means that the AUC prediction quality did not decrease as a result of using the new simultaneous method. The variation in training error across the methods was also small, about 2%. On the other hand, as expected, the graph traversal costs varied widely over the different methods and settings of C_1, as a result of the decrease in the probability estimates, as shown in Figure 10. There is a range of realistic probability estimates, but towards the right of the plot (for instance when $C_1 > .85$), the probability estimates are probably too low to be realistic and the costs are substantially underestimated. Let us compare the values of Cost 1 for our experiments in the more realistic range: Cost 1 of the naïve method was 24.5% higher than that of the sequential method; As C_1 was increased from 0.05 to 0.5, Cost 1 went from 27.5 units to 3.2 units, which is over eight times smaller. This means that with a 1-2% variation in the predictive model's AUC, the graph traversal cost can decrease a lot, potentially yielding a more cost-effective route for inspection and/or repair work, by favoring the cost to be underestimated when there is uncertainty.

In most applications relevant to this problem, we suspect that the solution used in practice is somewhere in between the naïve route and the sequential route, in that a human views the naïve solution and adjusts it by hand to be closer to the sequential route (without solving the TRP). For the application to electrical grid maintenance, the simultaneous

method was able to find a substantially lower cost route that the naïve or sequential method, with little (if any) change in the AUC prediction quality.

5. Generalization Bound

We initially introduced the graph traversal cost regularization term in order to find scenarios where the data would support low-cost (more actionable) repair routes. From another point of view, incorporating regularization reduces the size of the hypothesis space and may thus promote generalization. The size of the hypothesis space can be controlled using C_1. Increasing C_1 may thus assist in predicting failure probabilities on future inputs x using $f_\lambda(x)$. Here, it is irrelevant whether a new instance x is incorporated into a physical underlying graph, we aim only to estimate $P(y = 1|x)$, where all $\{x_i, y_i\}_{i=1}^m$ and new point x, y are chosen independently at random from unknown distribution $\mu_{\mathcal{X}\times\mathcal{Y}}$. In what follows, we will provide a generalization bound for the ML&TRP algorithm (22) with Cost 1 using an upper bound on Cost 1 to limit the size of the hypothesis space. A similar bound for (23) might be derived using the same method.

Generalization bounds are probabilistic guarantees that are useful for showing what variables may be important in the generalization process (Bousquet, 2003). The vast majority of works on generalization analysis are mainly interested in problems where the dimensionality d of the input space (or feature space) is very large, leading to the "curse of dimensionality." In that case, various measures of the complexity of the hypothesis space are incorporated into the bounds; these complexity measures can often gauge the richness of a class of functions in a way that is independent of the input dimension d. Examples of such measures include the VC dimension for $\{0, 1\}$-valued function classes, ϵ-fat shattering dimension for real valued functions, Rademacher complexity, and certain kinds of covering numbers (Vapnik, 1998; Bartlett and Mendelson, 2002; Mendelson and Vershynin, 2003; Zhang, 2002; Shawe-Taylor and Cristianini, 2002; Kolmogorov and Tikhomirov, 1959). In the present work, we are instead interested in how the graph traversal cost influences generalization for a fixed d, that is, we are interested in how C_1 affects generalization, and not so much interested in the dependence on d. We make the assumption that all input features affect prediction ability. This means that our bound will depend on the dimensionality of the input space d, and that there is no standard complexity measure that can reduce the complexity of the class of functions. Having this dependence on d is not uncommon; for example, covering number bounds depending on the "Pollard dimension" (equal to input dimension d when finite) have been obtained for bounded real-valued functions (see Theorem 14.21 of Anthony and Bartlett, 1999, Theorem 6 of Haussler, 1992). There are also many bounds that rely directly on the number of elements within the hypothesis space, for finite hypothesis spaces.

We are seeking to bound the true risk

$$R(f_\lambda) := E_{(x,y)\sim\mathcal{X}\times\mathcal{Y}} l_f(x, y) = \int \ln\left(1 + e^{-yf(x)}\right) \partial\mu_{\mathcal{X}\times\mathcal{Y}}(x, y),$$

where $l_f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, l_f is the logistic loss. We will bound $R(f_\lambda)$ by the empirical risk:

$$R(f_\lambda, \{x_i, y_i\}_1^m) = \frac{1}{m}\sum_{i=1}^m l_f(x_i, y_i) = \frac{1}{m}\sum_{i=1}^m \ln\left(1 + e^{-y_i f(x_i)}\right)$$

plus a complexity term. The complexity term takes into account the limitations on the hypothesis space of f, namely that $f \in \mathcal{F}$ where:

$$\mathcal{F} := \{f : f(x) = \lambda \cdot x \text{ for some } \lambda \in \mathbb{R}^d \text{ such that } ||\lambda||_2 \leq M_1\}.$$

Replacing the Lagrange multiplier C_1 in (22) with an explicit constraint, f is subject to the graph traversal cost constraint:

$$\min_{\pi} \sum_{i=1}^{M} \frac{1}{1 + e^{-f(\tilde{x}_{\pi(i)})}} L_\pi(\pi(i)) \leq C_g.$$

where C_g is a constant (inversely related to C_1).

We assume that the features are bounded, specifically, $x \in \mathcal{X} \subset \mathbb{R}^d$ with $\sup_{x \in \mathcal{X}} ||x||_2 \leq M_2$. We know $f_\lambda : \mathcal{X} \to [-M_1 M_2, M_1 M_2]$ by the Cauchy-Schwarz inequality since $\forall f_\lambda \in \mathcal{F}, \forall x \in \mathcal{X}, |f_\lambda(x)| \leq M_1 M_2$.

Let us define the set of functions that are subject to a constraint on the graph traversal cost:

$$\begin{aligned} \mathcal{F}_0 &:= \left\{ f : f \in \mathcal{F}, \min_{\pi \in \Pi} \sum_{i=1}^{M} L_\pi(\pi(i)) \frac{1}{1 + e^{-f(\tilde{x}_{\pi(i)})}} \leq C_g \right\} \\ &= \left\{ f : f \in \mathcal{F}, \min_{\pi \in \Pi} \sum_{i=1}^{M} L_\pi(i) \frac{1}{1 + e^{-f(\tilde{x}_i)}} \leq C_g \right\}, \end{aligned}$$

where recall $L_\pi(\pi(i))$, defined in (4) is the latency of the node $\pi(i)$, which is the cumulative distance traveled on a tour before reaching $\pi(i)$. Our goal in this section will be to show that a bound on the complexity of the class $\mathcal{F}_0$ may assist generalization.

Define d_i to be the shortest distance from the starting node to node i. Here, d_1 is the length of the shortest tour that visits all the nodes and returns to node 1. This means $d_i \leq L_\pi(i)$, and this inequality can be tight if the graph can be embedded into 1-dimensional Euclidean space (on a line). In what follows, we will fix a vector c, defined element-wise by:

$$c^j = \frac{\tilde{c}^j}{C_g - \tilde{c}_0}$$

where

$$\tilde{c}^j = \frac{e^{M_1 M_2}}{(1 + e^{M_1 M_2})^2} \left(\sum_i d_i \tilde{x}_i^j \right)$$

and

$$\tilde{c}_0 = \left(M_1 M_2 \frac{e^{M_1 M_2}}{(1 + e^{M_1 M_2})^2} + \frac{1}{1 + e^{M_1 M_2}} \right) \sum_i d_i.$$

It will be important that the vector c depends on C_g.

The main result follows from these definitions:

Theorem 1 ***(Main Result)*** *Let $\mathcal{X} = \{x \in \mathbb{R}^d : ||x||_2 \leq M_2\}$, $\mathcal{Y} = \{-1, 1\}$. Let $\mathcal{F}_0$ be defined as above with respect to $\{\tilde{x}_i\}_{i=1}^M$, $\tilde{x}_i \in \mathcal{X}$ (not necessarily random). Let $\{x_i, y_i\}_{i=1}^m$ be a sequence of m examples drawn independently according to an unknown distribution $\mu_{\mathcal{X}\times\mathcal{Y}}$. Then for any $\epsilon > 0$,*

$$P\Big(\exists f \in \mathcal{F}_0 : |R(f_\lambda, \{x_i, y_i\}_1^m) - R(f_\lambda)| > \epsilon\Big) \leq 4\alpha(d, C_g, c)\left(\frac{32M_1M_2}{\epsilon} + 1\right)^d \exp\left(\frac{-m\epsilon^2}{512(M_1M_2)^2}\right),$$

where

$$\alpha(d, C_g, c) := \frac{1}{2} + \frac{||c||_2^{-1} + \frac{\epsilon}{32M_2}}{M_1 + \frac{\epsilon}{32M_2}} \frac{\Gamma\left[1 + \frac{d}{2}\right]}{\sqrt{\pi}\Gamma\left[\frac{d+1}{2}\right]} {}_2F_1\left(\tfrac{1}{2}, \tfrac{1-d}{2}; \tfrac{3}{2}; \left(\frac{||c||_2^{-1} + \frac{\epsilon}{32M_2}}{M_1 + \frac{\epsilon}{32M_2}}\right)^2\right) \tag{25}$$

or equivalently

$$\alpha(d, C_g, c) := 1 - \frac{1}{2} I_{1-\left(||c||_2^{-1} + \frac{\epsilon}{32M_2}\right)^2 / \left(M_1 + \frac{\epsilon}{32M_2}\right)^2}\left(\frac{d+1}{2}, \frac{1}{2}\right) \tag{26}$$

and where ${}_2F_1(a, b; c; d)$ and $I_x(a, b)$ are the hypergeometric function and the regularized incomplete beta functions respectively.

The term $\alpha(d, C_g, c)$ comes directly from formulas for the volumes of spherical caps. Our goal was to establish that generalization can depend on C_g. The value of C_g enters into the bound through vector c. As C_g decreases, the norm $\|c\|_2$ increases, and thus $\|c\|_2^{-1}$ decreases, (26) and (25) decrease, and the whole bound decreases. This indicates that decreasing C_g may improve generalization ability.

We will provide several lemmas leading to the proof of the theorem. The proof idea is to enlarge the class of functions just enough so that a bound on the covering number (the number of balls required to cover the set $\mathcal{F}_0$) can be constructed. The class $\mathcal{F}$ corresponds to a ball of radius M_1 in λ-space (a ball in $\mathbb{R}^d$). The class $\mathcal{F}_0$ corresponds to a subset of that class. We will construct two classes, $\mathcal{F}_1$ and $\mathcal{F}_2$ that are slightly larger than $\mathcal{F}_0$, but smaller than $\mathcal{F}$ when C_g is small enough. Then we will use a volumetric argument to bound the covering number of $\mathcal{F}_2$, which uses the volumes of spherical caps. The idea is to show that the value of C_g affects the volume of the hypothesis space, and thus the covering number. The covering number bound is then applied to a uniform bound of Pollard (1984). The fact that the covering number of $\mathcal{F}_2$ can be below that of $\mathcal{F}$ indicates that using functions from $\mathcal{F}_2$ may provide improvements in generalization over the set $\mathcal{F}$. We now proceed with the proof.

We define the ball $\mathcal{F}_1$, using a lower bound on the latencies $L_\pi(i)$, namely the minimum distances d_i. We have, for any values of $p(\tilde{x}_i) \geq 0$:

$$\sum_i d_i p(\tilde{x}_i) \leq \sum_i L_\pi(i) p(\tilde{x}_i) \leq C_g.$$

This means that the class of functions who probabilities obey $\sum_i d_i p(\tilde{x}_i) \leq C_g$ is larger than the class obeying $\sum_i L_\pi(i) p(\tilde{x}_i) \leq C_g$. That is, $\mathcal{F}_0 \subseteq \mathcal{F}_1$ where

$$\mathcal{F}_1 := \left\{ f : f \in \mathcal{F}, \sum_{i=1}^M d_i \frac{1}{1 + e^{-f(\tilde{x}_{\pi(i)})}} \leq C_g \right\}.$$

As long as $C_g \leq \sum_{i=1}^{M} d_i$, the constraint in $\mathcal{F}_1$ is not vacuous.

Let $B_{M_1} = \{\lambda : ||\lambda||_2 \leq M_1\}$ be a closed ball of radius M_1 in $\mathbb{R}^d$. By definition, the set B_{M_1} are the set of λ used for constructing functions $\mathcal{F}$.

The space of functions $\mathcal{F}_2$ is defined with respect to the vector c (defined above the theorem) as follows:

$$\mathcal{F}_2 := \{f_\lambda : f_\lambda \in \mathcal{F}, c \cdot \lambda \leq 1\}.$$

The choice of c ensures that $\mathcal{F}_1$ is a subset of $\mathcal{F}_2$ as we will prove below. The half space corresponding to $\mathcal{F}_2$ is

$$H_{||c||_2^{-1}} := \{\lambda : c \cdot \lambda \leq 1\}.$$

The value $||c||_2^{-1}$ will be used in the volumetric argument.

We provide some common definitions.

Definition 2 *Let $A \subseteq X$ be an arbitrary set and $(X, dist)$ a (pseudo) metric space. Let $|\cdot|$ denote set size.*

- *For any $\epsilon > 0$, an ϵ-cover for A is a finite set $U \subseteq X$ (not necessarily $\subseteq A$) s.t. $\forall x \in A, \exists u \in U$ with $dist(x, u) \leq \epsilon$.*
- *A is totally bounded if A has a finite ϵ-cover for all $\epsilon > 0$. The covering number of A is $N(\epsilon, A, dist) := \inf_U |U|$ where U is an ϵ-cover for A.*
- *A set $R \subseteq X$ is ϵ-separated if $\forall x, y \in R$, $dist(x, y) > \epsilon$. The packing number $M(\epsilon, A, dist) := \sup_{R:R\subseteq A} |R|$, where R is ϵ-separated.*

There is a well-known relationship between packing numbers and covering numbers which we will make use of in proving Theorem 7.

Lemma 3 ***(Packing and covering numbers)*** *For every (pseudo) metric space $(X, dist)$, $A \subseteq X$, and $\epsilon > 0$,*

$$N(\epsilon, A, dist) \leq M(\epsilon, A, dist).$$

Proof See Theorem 4 in Kolmogorov and Tikhomirov (1959) or Theorem 12.1 in Anthony and Bartlett (1999) for a proof of this classical result. ■

Let $\mu_{\mathcal{B}}$ represent a probability measure on a set $\mathcal{B}$. Let B be a random variable taking values in $\mathcal{B}$ according to $\mu_{\mathcal{B}}$, and b be a realization of B. Let $\mu_{\mathcal{B}}^m$ represent the empirical measure based on sample $B_1^m = \{b_1, ..., b_m\}$. Let $L_2(\mu_{\mathcal{B}})$ be a space of functions defined on set $\mathcal{B}$ with the metric $\|f - g\|_{L_2(\mu_{\mathcal{B}})} = \int (f(b) - g(b))^2 d\mu_{\mathcal{B}}$. In what follows, we will define $\mathcal{B}$ to be the input space $\mathcal{X}$ or the joint input output space $\mathcal{X} \times \mathcal{Y}$. Also the squared ℓ_2 distance will continue to be denoted $\sum_j (\lambda_1^j - \lambda_2^j)^2 = \|\lambda_1 - \lambda_2\|_2^2$.

Lemma 4 ***(Relating covering numbers in $\|\cdot\|_{L_2(\mu_{\mathcal{X}}^m)}$ to $\|\cdot\|_2$)***

a. $\sup_{\mu_{\mathcal{X}}^m} N(\epsilon, \mathcal{F}, \|\cdot\|_{L_2(\mu_{\mathcal{X}}^m)}) \leq N(\epsilon/M_2, B_{M_1}, \|\cdot\|_2)$

b. $\sup_{\mu_{\mathcal{X}}^m} N(\epsilon, \mathcal{F}_2, \|\cdot\|_{L_2(\mu_{\mathcal{X}}^m)}) \leq N(\epsilon/M_2, B_{M_1} \cap H_{||c||_2^{-1}}, \|\cdot\|_2)$.

Lemma 4 will be used to tie together two results in the proof of the Theorem 1 to relate the covering number of a class of functions to the covering number of a subset of λ in $\mathbb{R}^d$. The first result is Lemma 5 which shows how the covering number of different function classes of interest are related. The second result is Theorem 7 which shows how covering number of subsets of $\mathbb{R}^d$ can be bounded.

Proof Each element $f \in \mathcal{F}$ corresponds to at least one element of B_{M_1} by definition of $\mathcal{F}$. Choose any distribution $\mu_{\mathcal{X}}^m$. Consider two elements $\lambda_f, \lambda_g \in B_{M_1}$ corresponding to functions $f, g \in \mathcal{F} \subset L_2(\mu_{\mathcal{X}}^m)$. Then,

$$\begin{aligned}
\|f-g\|^2_{L_2(\mu_{\mathcal{X}}^m)} &= \frac{1}{m}\sum_{i=1}^m (f(x_i)-g(x_i))^2 \\
&= \frac{1}{m}\sum_{i=1}^m ((\lambda_f-\lambda_g)\cdot x_i)^2 \\
&\leq \frac{1}{m}\sum_{i=1}^m \|\lambda_f-\lambda_g\|_2^2\|x_i\|_2^2 \text{ (Cauchy-Schwarz to each term)} \\
&\leq \|\lambda_f-\lambda_g\|_2^2\left(\frac{1}{m}\sum_{i=1}^m M_2^2\right) \text{ (since } \sup_{x\in\mathcal{X}}\|x\|_2 \leq M_2) \\
&= \|\lambda_f-\lambda_g\|_2^2 M_2^2.
\end{aligned}$$

Consider a minimal ϵ/M_2-cover $\{\lambda_r\}_r$ for B_{M_1} where λ_r corresponds to function $r \in \mathcal{F}$. Then by definition, $\forall\lambda \in B_{M_1}, \exists\lambda_r : \|\lambda-\lambda_r\|_2 \leq \epsilon/M_2$. Thus, picking any two such elements λ_f, λ_g in a ball of radius ϵ/M_2 around λ_r, we see that, the corresponding functions f, g belong to a ball of radius ϵ measured using distance in $L_2(\mu_{\mathcal{X}}^m)$ by the inequality above. The centers of these ϵ-balls in $L_2(\mu_{\mathcal{X}}^m)$ form an ϵ-cover for $\mathcal{F}$. The size of this set is equal to $N(\epsilon/M_2, B_{M_1}, \|\cdot\|_2)$. The size of the minimal ϵ-cover of $\mathcal{F}$ is less than or equal to this size, $N(\epsilon, \mathcal{F}, \|\cdot\|_{L_2(\mu_{\mathcal{X}}^m)}) \leq N(\epsilon/M_2, B_{M_1}, \|\cdot\|_2)$. Taking a supremum over all $\mu_{\mathcal{X}}^m$, we obtain the first inequality of the Lemma. The same argument also works for the second inequality. ■

We will upper bound the covering number for $\mathcal{F}_1$ with the covering number for the more tractable $\mathcal{F}_2$. We need to derive the vector c in such a way that $\mathcal{F}_2$ is a larger class than $\mathcal{F}_1$.

Lemma 5 *($\mathcal{F}_0$ **is contained in** $\mathcal{F}_2$)*

$$N(\epsilon, \mathcal{F}_0, \|\cdot\|_{L_2(\mu_{\mathcal{X}}^m)}) \leq N(\epsilon, \mathcal{F}_1, \|\cdot\|_{L_2(\mu_{\mathcal{X}}^m)}) \leq N(\epsilon, \mathcal{F}_2, \|\cdot\|_{L_2(\mu_{\mathcal{X}}^m)}).$$

Proof It is sufficient to show $\mathcal{F}_0 \subseteq \mathcal{F}_1 \subseteq \mathcal{F}_2$. The first inequality was discussed earlier; since $d_i \leq \inf_{\pi\in\Pi} L_\pi(i)$, this implies:

$$\sum_{i=1}^M d_i p(\tilde{x}_i) \leq \sum_{i=1}^M L_\pi(i) p(\tilde{x}_i) \leq C_g \Rightarrow \mathcal{F}_0 \subseteq \mathcal{F}_1.$$

We now show $\mathcal{F}_1 \subseteq \mathcal{F}_2$. We will show how $\tilde{c}$ and $\tilde{c}_0$ were derived so that:

$$\tilde{c} \cdot \lambda + \tilde{c}_0 \leq \sum_{i=1}^{M} d_i p(\tilde{x}_i) \leq \sum_{i=1}^{M} L_\pi(i) p(\tilde{x}_i) \leq C_g,$$

which will allow us to say that the set of λ such that $\tilde{c} \cdot \lambda + \tilde{c}_0 \leq C_g$ is larger than $\mathcal{F}_1$; this set is $\mathcal{F}_2$. We will find $\tilde{c}$ and $\tilde{c}_0$ by finding m_1 and m_0 such that for any i,

$$m_1(\lambda \cdot \tilde{x}_i) + m_0 \leq p(\tilde{x}_i), \tag{27}$$

so that $\tilde{c}$ and $\tilde{c}_0$ will be defined with respect to m_0 and m_1 by:

$$\sum_i d_i p(\tilde{x}_i) \geq \sum_i d_i (m_1(\lambda \cdot \tilde{x}_i) + m_0) = m_1 \left(\sum_i d_i \tilde{x}_i \right) \cdot \lambda + m_0 \sum_i d_i =: \tilde{c} \cdot \lambda + \tilde{c}_0. \tag{28}$$

Let us now define m_1 and m_0 in order to obey (27). The condition in (27) is:

$$m_1 f(\tilde{x}_i) + m_0 \leq \frac{1}{1 + e^{-f(\tilde{x}_i)}}.$$

Within the range $[-M_1 M_2, M_1 M_2]$ we lower bound the function $g(z) = 1/(1 + e^{-z})$ by the line with slope

$$m_1 = g'(-M_1 M_2) = \frac{e^{M_1 M_2}}{(1 + e^{M_1 M_2})^2}$$

that intersects the point $(-M_1 M_2, g(-M_1 M_2))$, and thus has y-intercept

$$m_0 = M_1 M_2 \frac{e^{M_1 M_2}}{(1 + e^{M_1 M_2})^2} + \frac{1}{1 + e^{M_1 M_2}}.$$

Incorporating this into (28), we have $\tilde{c} \cdot \lambda + \tilde{c}_0 \leq \sum_i d_i p(\tilde{x}_i) \leq C_g$, where $\tilde{c}$ is defined element wise by

$$\tilde{c}^j = m_1 \left(\sum_i d_i \tilde{x}_i^j \right) = \frac{e^{M_1 M_2}}{(1 + e^{M_1 M_2})^2} \left(\sum_i d_i \tilde{x}_i^j \right)$$

and

$$\tilde{c}_0 = m_0 \sum_i d_i = \left(M_1 M_2 \, \frac{e^{M_1 M_2}}{(1 + e^{M_1 M_2})^2} + \frac{1}{1 + e^{M_1 M_2}} \right) \sum_i d_i.$$

Finally, we obtain vector c from $\tilde{c}$ and $\tilde{c}_0$,

$$\begin{aligned} \{\lambda : \tilde{c} \cdot \lambda + \tilde{c}_0 \leq C_g\} &= \{\lambda : \tilde{c} \cdot \lambda \leq C_g - \tilde{c}_0\} \\ &= \{\lambda : \frac{\tilde{c}}{C_g - \tilde{c}_0} \cdot \lambda \leq 1\} \\ &=: \{\lambda : c \cdot \lambda \leq 1\} \end{aligned}$$

where c is defined element-wise by

$$c^j = \frac{\tilde{c}^j}{C_g - \tilde{c}_0}.$$

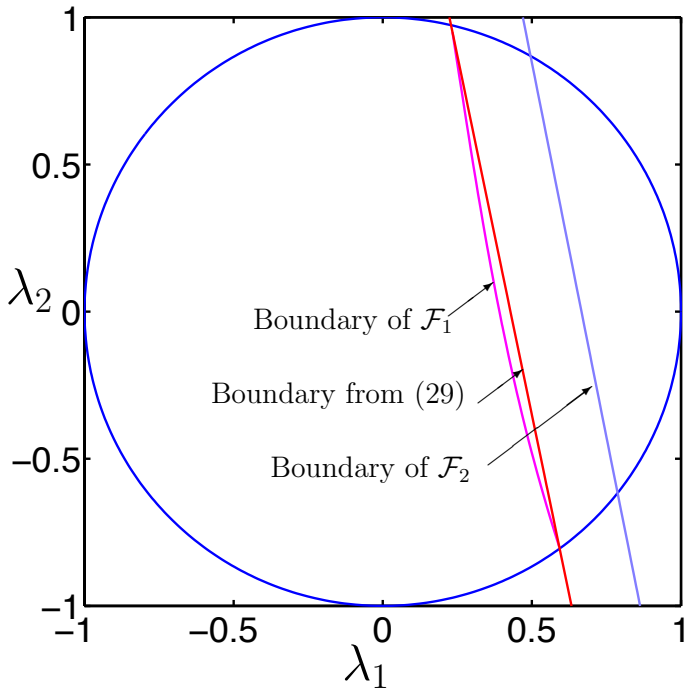


Figure 14: Hyperplanes upper bounding the nonlinear constraint in the description of $\mathcal{F}_1$. Here, $\lambda \in \mathbb{R}^2$. The ℓ_2 ball represents $\mathcal{F}$. For convenience, we have assumed $M_1 = 1$.

This is same the definition of c used in the theorem and for $\mathcal{F}_2$. Thus, $\mathcal{F}_2 \supseteq \mathcal{F}_1$. ■

Another way to obtain a suitable c such that $\mathcal{F}_2$ is a good superset of $\mathcal{F}_1$ is to minimize the distance of the hyperplane we want to construct from the origin by solving a semi-infinite program 29:

$$\begin{aligned} &\max_c ||c||_2^2 \\ &\text{s.t. } \forall \lambda \in \Lambda \cup \Lambda^0, c \cdot \lambda \leq 1 \\ &\text{where } \Lambda = \{\lambda : \lambda \in B_{M_1}, \sum_{i=1}^{M} d_i \frac{1}{1+\exp\left(-\lambda \cdot \tilde{x}_i\right)} = C_g\} \\ &\text{and } \Lambda^0 = \{\lambda : ||\lambda||_2 = M_1, \sum_{i=1}^{M} d_i \frac{1}{1+\exp\left(-\lambda \cdot \tilde{x}_i\right)} \leq C_g\}. \end{aligned} \tag{29}$$

One can approximate the two sets Λ and Λ^0 in the program formulation by discretizing the points on them, and the semi-infinite program becomes a non-linear program. Figure (14) provides a 2-dimensional illustration of $\mathcal{F}, \mathcal{F}_1, \mathcal{F}_2$ and the approximate solution of the semi-infinite program. The semi-infinite program yields a tighter bound on $\mathcal{F}_1$ but is more expensive to compute.

Because of rotational symmetry of B_{M_1}, the volume cut off by a hyperplane $c \cdot \lambda = 1$ from B_{M_1} is determined only by its distance from the origin, which is $1/||c||_2$. Such a portion (or its complement, if smaller) of a ball obtained from slicing it with a hyperplane is called a spherical cap. It can be parameterized by the distance of its (hyper)plane base from the center of the ball.

Let the volume of a set $A \subset \mathbb{R}^d$ be represented as $Vol(A)$. For example, $Vol(B_1) = \frac{\pi^{d/2}}{\Gamma[d/2+1]}$.

Lemma 6 *(**Volume of spherical caps**) Let the volume of ball B_{M_1} in $\mathbb{R}^d$ be denoted as $Vol(B_{M_1})$. Let $H_z = \{\lambda : c \cdot \lambda \leq 1, ||c||_2^{-1} = z\}$ be a half space parameterized by z. Let the spherical cap be denoted by $B_{M_1} \cap H_z'$ where the cap is at a distance z (measured from the base of the cap to the center of the ball), and H_z' represents the complement half space ($H_z \cup H_z' = \mathbb{R}^d$). Then,*

$$Vol(B_{M_1} \cap H_z') \quad = Vol(B_{M_1}) \left(\frac{1}{2} - \frac{z}{M_1} \frac{\Gamma\left[1+\frac{d}{2}\right]}{\sqrt{\pi}\Gamma\left[\frac{d+1}{2}\right]} {}_2F_1\left(\frac{1}{2}, \frac{1-d}{2}; \frac{3}{2}; \left(\frac{z}{M_1}\right)^2 \right) \right)$$

where ${}_2F_1(a, b; c; d)$ is the hypergeometric function. Alternatively,

$$Vol(B_{M_1} \cap H_z') \quad = Vol(B_{M_1}) \frac{1}{2} I_{1-z^2/M_1^2}\left(\frac{d+1}{2}, \frac{1}{2}\right)$$

where $I_x(e, f)$ is the regularized incomplete beta function.

Proof See Li (2011) and references therein. ■

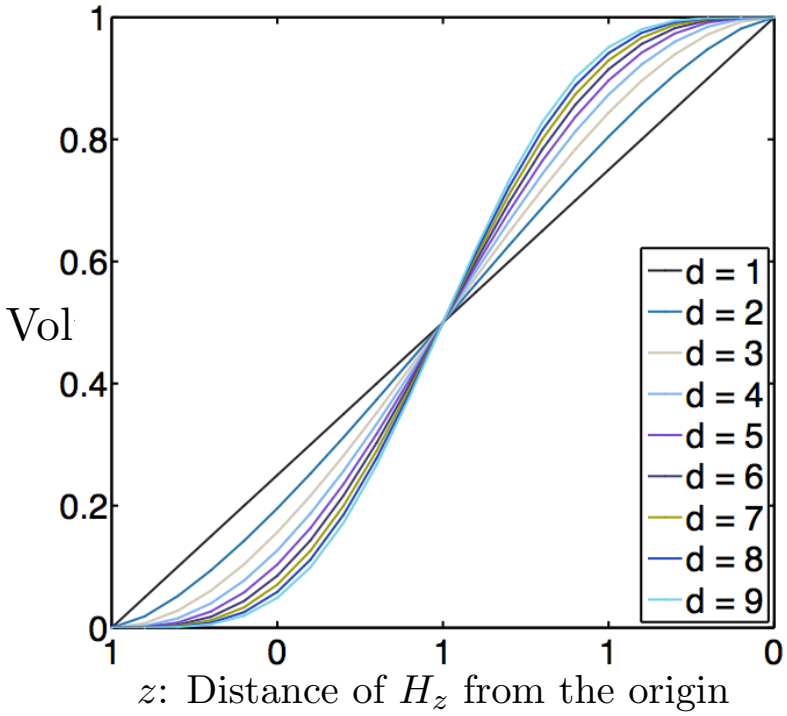


Figure 15: Normalized volume of a unit ℓ_2-ball intersected with a halfspace ($Vol(B_1 \cap H_z)$), as a function of the distance of the hyperplane from the center of the ball. This also illustrates the dependence of $\alpha(d, C_g, c)$ on C_g.

Note that for $0 \leq z \leq M_1, Vol(B_{M_1} \cap H'_z) \leq Vol(B_{M_1})$. If $Vol(B_{M_1} \cap H'_z) \leq \frac{1}{2} Vol(B_{M_1})$, then the volume of the spherical cap reduces with increasing dimension d. This has an important effect on the covering number as a function of dimension d, and ultimately on generalization too. Figure 15 illustrates this point by showing the volume on one side of the hyperplane as the hyperplane moves through the ball, for various values of the dimension d. If d is fairly large, then the volume decreases dramatically as the hyperplane passes through the center of the ball.

We now use the volume of the spherical cap in Lemma 6 to bound the covering numbers of subsets of $\mathbb{R}^d$, noting that the relationship between the spherical cap and its complement is: $Vol\left(B_{M_1} \cap H'_z\right) = Vol(B_{M_1}) - Vol\left(B_{M_1} \cap H_z\right)$.

Theorem 7 ***(Bound on Covering Numbers)***

$$
\begin{aligned}
N(\epsilon/M_2, B_{M_1}, \|\cdot\|_2) &\leq \left(\frac{2M_1M_2}{\epsilon} + 1\right)^d \\
N\left(\epsilon/M_2, B_{M_1} \cap H_{||c||_2^{-1}} \|\cdot\|_2\right) &\leq \left(\frac{Vol\left(B_{M_1+\frac{\epsilon}{2M_2}} \cap H_{||c||_2^{-1}+\frac{\epsilon}{2M_2}}\right)}{Vol\left(B_{M_1+\frac{\epsilon}{2M_2}}\right)}\right)\left(\frac{2M_1M_2}{\epsilon} + 1\right)^d.
\end{aligned}
$$

Proof Both statements involve a volumetric argument. There are various versions of proof for the first part. For example, see Section 3 of Kolmogorov and Tikhomirov (1959), Lemma 4.10 in Pisier (1989), Lorentz (1966) and Lemma 3 in Cucker and Smale (2002) among others. We will provide an argument along these lines. Let $\lambda_1, .., \lambda_M$ be an optimal ϵ-packing for B_{M_1}. That is, $M = M(\epsilon, B_{M_1}, \|\cdot\|_2)$. The volume of $B_{M_1+\epsilon/2}$ (an extra $\epsilon/2$ added so that the packing elements can lie within the boundary) is,

$$Vol(B_{M_1+\epsilon/2}) = Vol(B_1)(M_1 + \epsilon/2)^d$$

where $Vol(B_1)$ is the volume of a unit ball in dimension d. The volume of an $\epsilon/2$ ball with packing element λ_i as the center is:

$$Vol(B_{\epsilon/2} + \lambda_i) = Vol(B_{\epsilon/2}) = Vol(B_1)(\epsilon/2)^d.$$

Since the sum of the volume of the $\epsilon/2$ balls should be less than or equal to the volume of the extended ball $B_{M_1+\epsilon/2}$ (else one of the packing elements λ_i will be outside the boundary of B_{M_1} contradicting the definition of packing) we have:

$$M(\epsilon, B_{M_1}, \|\cdot\|_2)Vol(B_1)(\epsilon/2)^d \leq Vol(B_1)(M_1 + \epsilon/2)^d.$$

Scaling ϵ to ϵ/M_2 and using the inequality between minimal covering and maximal packing numbers from Lemma 3 we obtain the first stated result.

To show the second part, let the volume of the complement of the spherical cap be $Vol(B_{M_1} \cap H_{||c||_2^{-1}})$; we need to find an upper bound for the minimal ϵ/M_2-cover of this set. We can do that by scaling a minimal ϵ-cover, which we find now. By extending the boundary of $B_{M_1} \cap H_{||c||_2^{-1}}$ by $\epsilon/2$ we can bound the maximal packing number $M(\epsilon, B_{M_1} \cap H_{||c||_2^{-1}}, \|\cdot\|_2)$ as follows.

$$M(\epsilon, B_{M_1} \cap H_{||c||_2^{-1}}, \|\cdot\|_2)Vol(B_1)(\epsilon/2)^d \quad \leq \quad Vol(B_{M_1+\epsilon/2} \cap H_{||c||_2^{-1}+\epsilon/2})$$

$$\begin{aligned}
M(\epsilon, B_{M_1} \cap H_{||c||_2^{-1}}, \|\cdot\|_2) &\leq \left(\frac{Vol\left(B_{M_1+\epsilon/2} \cap H_{||c||_2^{-1}+\epsilon/2}\right)}{Vol(B_1)}\right) \frac{1}{(\epsilon/2)^d} \\
&= \left(\frac{Vol\left(B_{M_1+\epsilon/2} \cap H_{||c||_2^{-1}+\epsilon/2}\right)}{Vol(B_1)}\right) \frac{1}{\epsilon/2^d} \frac{(M_1 + \epsilon/2)^d}{(M_1 + \epsilon/2)^d} \\
&= \left(\frac{Vol\left(B_{M_1+\epsilon/2} \cap H_{||c||_2^{-1}+\epsilon/2}\right)}{Vol(B_{M_1+\epsilon/2})}\right) \frac{(M_1 + \epsilon/2)^d}{(\epsilon/2)^d}.
\end{aligned}$$

Again, scaling ϵ to ϵ/M_2 and using the relationship between $N(\epsilon, A, dist)$ and $M(\epsilon, A, dist)$ in Lemma 3 yields the second result. ■

We are now done with the covering number and volumetric arguments. What remains is to show how a uniform generalization bound can be adapted to handle covering numbers.

We initially concern ourselves with the class of loss functions $l_{\mathcal{F}} := \{l_f : f \in \mathcal{F}\}$, and adapt the bound to handle classes $\mathcal{F}$, $\mathcal{F}_2$ and $\mathcal{F}_0$. The form and proof of the convergence in terms of the size of $l_{\mathcal{F}}$ becomes simpler when each $l_f \in l_{\mathcal{F}}$ is non-negative and bounded. This is indeed the case since the logistic loss is non-negative and bounded (the latter because $\mathcal{F}$ and its subsets are bounded sets of linear functions). We will use the following uniform convergence bound of Pollard (1984).

Theorem 8 ***(Pollard 1984)*** *Let $l_{\mathcal{F}}$ be a set of functions on $\mathcal{X} \times \mathcal{Y}$ with $0 \leq l_f(x,y) \leq M_{bound}, \forall l_f \in l_{\mathcal{F}}$ and $\forall (x,y) \in \mathcal{X} \times \mathcal{Y}$. Let $\{x_i, y_i\}_1^m$ be a sequence of m examples drawn independently according to $\mu_{\mathcal{X}\times\mathcal{Y}}$. Then for any $\epsilon > 0$,*

$$P(\exists l_f \in l_{\mathcal{F}} : |R(f_\lambda, \{x_i, y_i\}_1^m) - R(f_\lambda)| > \epsilon) \leq 4E[N(\epsilon/16, l_{\mathcal{F}}, \|\cdot\|_{L_1(\mu^m_{\mathcal{X}\times\mathcal{Y}})})] \exp\left(\frac{-m\epsilon^2}{128 M_{bound}^2}\right).$$

Proof See Theorem 24 in Pollard (1984) (also in Zhang, 2002, Theorem 1). Note that the constants have been refined in other works since the first result and we have left the original constants intact here. ■

We can relate the covering numbers for Pollard's $l_{\mathcal{F}}$ and covering numbers for $\mathcal{F}$ as follows.

Lemma 9 ***(Relating $l_{\mathcal{F}}$ to $\mathcal{F}$)*** *If every function from function class $l_{\mathcal{F}}$ represented as $l : f(\mathcal{X}) \times \mathcal{Y} \mapsto \mathbb{R}, f \in \mathcal{F}$ is Lipschitz in its first argument with Lipschitz constant $\mathcal{L}$, then the covering number of $l_{\mathcal{F}}$ is related to the covering number of $\mathcal{F}$ as*

$$\sup_{\mu^m_{\mathcal{X}\times\mathcal{Y}}} N(\epsilon, l_{\mathcal{F}}, \|\cdot\|_{L_1(\mu^m_{\mathcal{X}\times\mathcal{Y}})}) \leq N(\epsilon/\mathcal{L}, \mathcal{F}, \|\cdot\|_{L_1(\mu^m_{\mathcal{X}})})$$

Proof Consider two functions $f, g \in \mathcal{F}$. Let the corresponding functions in class $l_{\mathcal{F}}$ be $l_f(x,y) = l(f(x), y)$ and $l_g = l(g(x), y)$.

$$\begin{aligned} \|l_f - l_g\|_{L_1(\mu^m_{\mathcal{X}\times\mathcal{Y}})} &= \frac{1}{m}\sum_{i=1}^m |l_f(x_i,y_i) - l_g(x_i,y_i)| = \frac{1}{m}\sum_{i=1}^m |l(f(x_i),y_i) - l(g(x_i),y_i)| \\ &\leq \frac{1}{m}\sum_{i=1}^m \mathcal{L}|f(x_i) - g(x_i)| = \mathcal{L}\|f - g\|_{L_1(\mu^m_{\mathcal{X}})}. \end{aligned}$$

This implies, given $\{X, Y\}_1^m$, if $\hat{\mathcal{F}}$ is a minimal $\epsilon/\mathcal{L}$-cover of $\mathcal{F}$ in $L_1(\mu^m_{\mathcal{X}})$, we can construct an ϵ-cover of $l_{\mathcal{F}}$ in $L_1(\mu^m_{\mathcal{X}\times\mathcal{Y}})$ as

$$\hat{l}_{\mathcal{F}} = \{l_{f_i} : f_i \in \hat{\mathcal{F}}.\}$$

■

The logistic loss $\log(1 + e^{-yf(x)})$ when viewed as a function of $f(x)$ has a Lipschitz constant $\mathcal{L} \leq 1$. For a similar result using the squared loss see Lemma 17.4 of Anthony and Bartlett (1999).

Theorem 8 involves an L_1 covering number, but our volumetric argument is in terms of an L_2 covering number. The following lemma applies the statement $\|f-g\|_{L_1(\mu_X^m)} \leq \|f-g\|_{L_2(\mu_X^m)}$ (true because of Jensen's inequality applied to norms) to the covering numbers.

Lemma 10 $N(\epsilon, A, \|\cdot\|_{L_1(\mu_{\mathcal{X}}^m)}) \leq N(\epsilon, A, \|\cdot\|_{L_2(\mu_{\mathcal{X}}^m)})$.

Proof See for a version, Lemma 10.5 in Anthony and Bartlett (1999). ■

Finally, we can prove the main result.

Proof *(Of Theorem 1)* Starting from the expectation term on the right hand side of Theorem 8,

$$
\begin{aligned}
& E[N(\epsilon/16, l_{\mathcal{F}_0}, \|\cdot\|_{L_1(\mu^m_{\mathcal{X}\times\mathcal{Y}})})] \\
& \leq \; E[N(\epsilon/16, l_{\mathcal{F}_2}, \|\cdot\|_{L_1(\mu^m_{\mathcal{X}\times\mathcal{Y}})})] \text{ from Lemma 5} \\
& \leq \; \sup_{\mu^m_{\mathcal{X}\times\mathcal{Y}}} N(\epsilon/16, l_{\mathcal{F}_2}, \|\cdot\|_{L_1(\mu^m_{\mathcal{X}\times\mathcal{Y}})}) \text{ bounding expectation by supremum} \\
& \leq \; \sup_{\mu^m_{\mathcal{X}}} N\left(\frac{\epsilon}{16\mathcal{L}}, \mathcal{F}_2, \|\cdot\|_{L_1(\mu^m_{\mathcal{X}})}\right) \text{ from Lemma 9} \\
& \leq \; \sup_{\mu^m_{\mathcal{X}}} N\left(\frac{\epsilon}{16\mathcal{L}}, \mathcal{F}_2, \|\cdot\|_{L_2(\mu^m_{\mathcal{X}})}\right) \text{ from Lemma 10} \\
& \leq \; N\left(\frac{\epsilon}{16\cdot 1\cdot M_2}, B_{M_1}\cap H_{\|c\|_2^{-1}}, \|\cdot\|_2\right) \text{ from Lemma 4 and substituting } \mathcal{L}=1 \\
& \leq \; \left(\frac{Vol\left(B_{M_1+\frac{\epsilon}{32M_2}}\cap H_{\|c\|_2^{-1}+\frac{\epsilon}{32M_2}}\right)}{Vol(B_{M_1+\frac{\epsilon}{32M_2}})}\right)\left(\frac{32M_1M_2}{\epsilon}+1\right)^d \text{ from Theorem 7} \\
& = \; \alpha(d, C_g, c)\left(\frac{32M_1M_2}{\epsilon}+1\right)^d \text{ from Lemma 6.}
\end{aligned}
$$

The above step uses the relationship between the spherical cap and its complement along with Lemma 6,

$$Vol\left(B_{M_1}\cap H'_{\|c\|_2^{-1}}\right) = Vol(B_{M_1}) - Vol\left(B_{M_1}\cap H_{\|c\|_2^{-1}}\right).$$

Using the bound on $E[N(\epsilon/16, l_{\mathcal{F}_0}, \|\cdot\|_{L_1(\mu^m_{\mathcal{X}\times\mathcal{Y}})})]$ obtained above in Theorem 8 gives the result. ■

6. Discussion and Related Works

In this work, we present a machine learning algorithm that takes into account the way its recommendations will be ultimately used. This algorithm takes advantage of uncertainty in the model in order to potentially find a much more practical solution. Including these

operating costs is a new way of incorporating "structure" into machine learning algorithms, and we plan to explore this in other ways in ongoing work. One main focus of the work is to discuss a tradeoff between training error and operational cost. In doing so, we showed a new way in which data dependent regularization can influence an algorithm's prediction ability, formalized through generalization bounds.

There is a vast literature on regularization, but in the past it has been used to impose prior beliefs (e.g., "structure" such as sparsity like Tibshirani, 1996, shrinking certain coefficients towards each other), robustness (e.g., to obtain a large "margin" which is the distance from the decision boundary to the nearest training example like Vapnik, 1998), or additional distributional information (semi-supervised learning, see for instance Chapelle et al., 2006). Out of these, only semi-supervised learning uses unlabeled data, but our problem differs in that our unlabeled data does not need to be drawn from the same distribution as the training data, and thus does not necessarily provide any distributional information. It provides instead information about the practical cost of following the algorithm's recommendations.

In addition to the above, we developed cost models that apply to routing problems. For the power grid application and other maintenance applications, $\{d_{ij}\}_{ij}$ in (4) correspond to physical distances. It is possible to use the techniques developed here for more abstract routing problems, for instance, network scheduling or network routing problems, where distance on the graph does not necessarily correspond to a physical distance. There are other works that schedule events based on a linearly increasing cost model (see for instance Anily et al., 1998).

There is a body of literature regarding cost models for maintenance in the reliability modeling literature, though the emphasis in those works is usually to design a model that accurately represents the stochastic process for the failures. In particular, there are works on condition-based maintenance, where a maintenance schedule is created from the predicted condition of the equipment (but not on the cost of performing the repairs in a certain order or routing a vehicle between the equipment). Barbera et al. (1996) develop a model that assumes that equipment have exponential rates of failure and fail only once in an inspection interval, and they use this model to determine a maintenance schedule. Marseguerra et al. (2002) introduces a model for degradation leading to failure for a continuous complex system, and use Monte Carlo simulations to determine the optimal degradation level to perform an inspection. Their work uses a very different cost model from ours; the cost is the long run average maintenance cost and cost of failures. A neural-network based maintenance model was developed by Heng et al. (2009). Another large body of work considers more sophisticated estimates for system faults: for example, by modeling (repeat) measurements as time series (Xu et al., 2009). Depending on the application, one could replace the training error in our model with a more elaborate failure model such as the ones developed in these works.

If we were able to find an efficient method for approximately solving the TRP subproblem, it could allow us to compute solutions to the ML&TRP significantly faster. Constant factor approximation algorithms for the standard (unweighted) TRP have been developed in several works (Goemans and Kleinberg, 1998; Blum et al., 1994; Arora and Karakostas, 2006; Archer et al., 2008; Archer and Blasiak, 2010). These schemes typically have (quasi-) polynomial time guarantees and approximate up to a constant ratio of the optimal standard TRP objective value. The constant factors are at least 3.59 or above. Heuristic methods

might also be used for solving the standard TRP and related problems (Dewilde et al., 2010; Salehipour et al., 2010), which can potentially be adapted to solve the weighted TRP. There are some difficulties in doing this because the heuristics depend on the exact way the cost is defined. For example, Dewilde et al. (2010) solve a variation of the TRP which cannot easily be adapted for solving the weighted TRP problem. Lechmann (2009) has a survey of the various applications and solution techniques of the different versions of TRP problem.

There could be many variations on the setup for the ML&TRP. In some applications, real time sensor measurements are available, and it is possible to automatically turn off the equipment when it fails in order to prevent more failures from occurring. This is not possible for the power grid application, since it is not possible (and not desirable) to turn off the electricity supply in the secondary electrical distribution network, but it may be possible in other applications.

A related work on routing for emergency maintenance on the electrical grid is the heuristic algorithm of Weintraub et al. (1999) that dispatches vehicles to areas where there are currently breakdowns and where there are likely to be breakdowns in the future.

Acknowledgements

This material is based upon work supported by an International Fulbright Science and Technology Award, the MIT Energy Initiative, and the National Science Foundation under Grant No IIS-1053407.

References

Shivani Agarwal. Ranking on graph data. In *Proceedings of the 23rd International Conference on Machine Learning*, 2006.

Shoshana Anily, Celia A. Glass, and Refael Hassin. The scheduling of maintenance service. *Discrete Applied Mathematics*, 82(1-3):27–42, 1998.

Martin Anthony and Peter L. Bartlett. *Neural network learning: Theoretical foundations*. Cambridge University Press, 1999.

Aaron Archer and Anna Blasiak. Improved approximation algorithms for the minimum latency problem via prize-collecting strolls. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 429–447, 2010.

Aaron Archer, Asaf Levin, and David P. Williamson. A faster, better approximation algorithm for the minimum latency problem. *SIAM J. Comput.*, 37(5):1472–1498, 2008.

Sanjeev Arora and George Karakostas. A $2 + \epsilon$ approximation algorithm for the k-MST problem. *Math. Program.*, 107(3):491–504, 2006.

Fran Barbera, Helmut Schneider, and Peter Kelle. A condition based maintenance model with exponential failures and fixed inspection intervals. *The Journal of the Operational Research Society*, 47(8):pp. 1037–1045, 1996.

Peter L. Bartlett and Shahar Mendelson. Gaussian and Rademacher complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

Avrim Blum, Prasad Chalasani, Don Coppersmith, Bill Pulleyblank, Prabhakar Raghavan, and Madhu Sudan. On the minimum latency problem. *ArXiv Mathematics e-prints*, September 1994.

Pierre Bonami, Lorenz T. Biegler, Andrew R. Conn, Gérard Cornuéjols, Ignacio E. Grossmann, Carl D. Laird, Jon Lee, Andrea Lodi, François Margot, Nicolas W. Sawaya, and Andreas Wächter. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization*, 5(2):186–204, 2008.

Olivier Bousquet. New approaches to statistical learning theory. *Annals of the Institute of Statistical Mathematics*, 55(2):371–389, 2003.

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

Imre Csiszár and G. Tusnády. Information geometry and alternating minimization procedures. *Statistics and Decisions*, 1(Suppl.):205–237, 1984.

Felipe Cucker and Steve Smale. On the mathematical foundations of learning. *Bulletin-American Mathematical Society*, 39(1):1–50, 2002.

Thijs Dewilde, Dirk Cattrysse, Sofie Coene, Frits C. R. Spieksma, and Pieter Vansteenwegen. Heuristics for the Traveling Repairman Problem with Profits. *10th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems*, pages 34–44, 2010.

C. A. Eijl van. A polyhedral approach to the delivery man problem. Technical report, Memorandum COSOR 95–19, Department of Mathematics and Computer Science, Eindhoven University of Technology, The Netherlands, 1995.

Imen Ome Ezzine, Fréderic Semet, and Habib Chabchoub. New formulations for the Traveling Repairman Problem. In *Proceedings of the 8th International Conference of Modeling and Simulation*, pages 1889–1894, May 2010.

Matteo Fischetti, Gilbert Laporte, and Silvano Martello. The delivery man problem and cumulative matroids. *Oper. Res.*, 41:1055–1064, November 1993.

Michel Goemans and Jon Kleinberg. An improved approximation ratio for the minimum latency problem. *Mathematical Programming*, 82:111–124, 1998.

David Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and computation*, 100(1):78–150, 1992.

Aiwina Heng, Andy C.C. Tan, Joseph Mathew, Neil Montgomery, Dragan Banjevic, and Andrew K.S. Jardine. Intelligent condition-based prediction of machinery reliability. *Mechanical Systems and Signal Processing*, 23(5):1600 – 1614, 2009.

Waltraud Huyer and Arnold Neumaier. Global optimization by multilevel coordinate search. *J. of Global Optimization*, 14:331–355, June 1999.

Andrey Nikolaevich Kolmogorov and Vladimir Mikhailovich Tikhomirov. ε-entropy and ε-capacity of sets in function spaces. *Uspekhi Matematicheskikh Nauk*, 14(2):3–86, 1959.

Miriam Lechmann. The traveling repairman problem - an overview. *Diplomarbeit, Universitat Wein*, pages 1–79, 2009.

Shengqiao Li. Concise Formulas for the Area and Volume of a Hyperspherical Cap. *Asian Journal of Mathematics & Statistics*, 4(1):66–70, 2011.

George G. Lorentz. Metric entropy and approximation. *Bull. Am. Math. Soc.*, 72:903–937, 1966.

Marzio Marseguerra, Enrico Zio, and Luca Podofillini. Condition-based maintenance optimization by means of genetic algorithms and monte carlo simulation. *Reliability Engineering & System Safety*, 77(2):151 – 165, 2002.

Shahar Mendelson and Roman Vershynin. Entropy and the combinatorial dimension. *Inventiones Mathematicae*, 152(1):37–55, 2003.

Isabel Méndez-Díaz, Paula Zabala, and Abilio Lucena. A new formulation for the traveling deliveryman problem. *Discrete Applied Mathematics*, 156(17):3223–3237, 2008.

John Ashworth Nelder and Roger Mead. A simplex method for function minimization. *Computer Journal*, 7(4):308–313, 1965.

Jean-Claude Picard and Maurice Queyranne. The time-dependent traveling salesman problem and its application to the tardiness problem in one-machine scheduling. *Operations Research*, 26(1):86–110, January–February 1978.

Gilles Pisier. *The volume of convex bodies and Banach space geometry*, volume 94. Cambridge University Press, Cambridge, 1989.

David Pollard. *Convergence of stochastic processes*. Springer, 1984.

Luis Miguel Rios. Algorithms for derivative-free optimization. *PhD thesis, University of Illinois at Urbana-Champaign*, pages 1–133, 2009.

Cynthia Rudin, Rebecca Passonneau, Axinia Radeva, Haimonti Dutta, Steve Ierome, and Delfina Isaac. A process for predicting manhole events in Manhattan. *Machine Learning*, 80:1–31, 2010.

Cynthia Rudin, David Waltz, Roger Anderson, Albert Boulanger, Ansaf Salleb-Aouissi, Maggie Chow, Haimonti Dutta, Phil Gross, Bert Huang, Steve Ierome, Delfina Isaac, Artie Kressner, Rebecca Passonneau, Axinia Radeva, and Leon Wu. Machine learning for the New York City power grid. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011. accepted subject to minor revision.

Amir Salehipour, Kenneth Sorensen, Peter Goos, and Olli Bräysy. Efficient GRASP+ VND and GRASP+ VNS metaheuristics for the traveling repairman problem. *4OR: A Quarterly Journal of Operations Research*, pages 1–21, 2010.

Robert E. Schapire and Yoav Freund. *Boosting: Foundations and Algorithms*. MIT Press, 2011. In Preparation.

John Shawe-Taylor and Nello Cristianini. On the generalization of soft margin algorithms. *IEEE Transactions on Information Theory*, 48(10):2721–2735, 2002.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996. ISSN 0035-9246.

Ian Urbina. Mandatory safety rules are proposed for electric utilities. *New York Times*, 2004. August 21, Late Edition, Section B, Column 3, Metropolitan Desk, Page 2.

Vladimir Naumovich Vapnik. *Statistical learning theory*, volume 2. Wiley New York, 1998.

Andrés Weintraub, J. Aboud, C. Fernandez, G. Laporte, and E. Ramirez. An emergency vehicle dispatching system for an electric utility in Chile. *Journal of the Operational Research Society*, pages 690–696, 1999.

Zhengguo Xu, Yindong Ji, and Donghua Zhou. A new real-time reliability prediction method for dynamic systems based on on-line fault prediction. *IEEE Transactions on Reliability*, 58(3):523–538, 2009.

Tong Zhang. Covering number bounds of certain regularized linear function classes. *Journal of Machine Learning Research*, 2:527–550, 2002.

Dengyong Zhou, Jason Weston, Arthur Gretton, Olivier Bousquet, and Bernhard Schölkopf. Ranking on data manifolds. In *Advances in Neural Information Processing Systems 16*, pages 169–176. MIT Press, 2004.