PREDICTION OF ARRIVAL TIMES OF FREIGHT TRAFFIC ON US RAILROADS USING SUPPORT VECTOR REGRESSION

BY

WILLIAM W. BARBOUR

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Civil and Environmental Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2017

Urbana, Illinois

Adviser:

Assistant Professor Daniel B. Work

ABSTRACT

Variability of the travel times on the United States freight rail network is high due to large network demand relative to infrastructure capacity especially when traffic is heterogeneous. Variable runtimes pose significant operational challenges if the nature of runtime variability is not predictable. To address this issue, this article proposes a data-driven approach to predict *estimated times of arrival* (ETAs) of individual freight trains, based on the properties of the train, the properties of the network, and the properties of potentially conflicting traffic on the network. The ETA problem is posed as a machine learning regression problem and solved using a support vector regression machine trained and cross validated on over two years of historical data for a 140 mile stretch of track located primarily in Tennessee, USA. The article presents the data used in this problem and details on feature engineering and construction for predictions made across the full route. It also highlights findings on the dominant sources of runtime variability and the most predictive factors for ETA, identified by applying the data framework. ETA improvement results exceeded 20% over baseline methods for predictions made at some locations and averaged over 15% across the study area. Ideas for further ETA improvement using the prediction algorithms are also discussed.

To my parents, for their love and support. And to all of those at the University of Illinois who have helped me with this chapter of my career.

ACKNOWLEDGMENTS

The author acknowledges funding by the Roadway Safety Institute, the University Transportation Center for USDOT Region 5, which Includes Minnesota, Illinois, Indiana, Michigan, Ohio, and Wisconsin. Financial support was provided by the United States Department of Transportation's Office of the Assistant Secretary for Research and Technology (OST-R). Thanks, also, to CSX Transportation for their expertise, support, and research collaboration.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Motivation

The rail network in the United States has significant infrastructure capacity limitations that cause congestion of the rail traffic. Few rail corridors contain exclusively double (or more) track that allows simultaneous bi-directional traffic [1]. In comparison, the double and triple track railroads in Europe provide for double the train density of US rail networks [2]. Many US corridors contain a single track with short sections of double track known as *sidings*, where trains may meet or pass each other. These *movements* (i.e., meets, passes) are implemented in the railroad signaling system, but are directed by human dispatchers. Dispatchers are experienced with working on specific track corridors, but movements on sidings require planning and precise timing in order to achieve efficient operations [3, 4]. Freight volume is expected to increase in the US, so either infrastructure capacity must be increased or operational improvements must be made to increase capacity [5, 6, 7].

In addition to the track infrastructure constraints, there are numerous other factors that can contribute to variability of the runtime on a track segment. Traffic heterogeneity and the train priority differences directly influence both the runtime of trains and also the variability in the runtime [8, 9]. Physical characteristics of trains such as the length, tonnage, and power further influence the runtime due to track grade, track curvature and siding lengths [8]. The ability of a train to complete a trip and exit the *line of road* (i.e., the track segments connecting distant terminals) is also influenced by the degree of congestion in the arrival terminal. This is compounded by the possible actions required for the train in the terminal, such as refueling, inspection, switching of cars, or crew change [8, 10]. Railroad operating strategies such as dynamically scheduled trains and maximizing train length

are particularly vulnerable to delay [11, 12].

In the presence of runtime variability, ETAs are necessary in order to improve real-time decision making and the efficiency of the network [13, 14]. For example, future train schedules can be continually updated to provide new train plans to allow traffic to flow smoothly between terminals on the network [15]. Although there are many techniques available to derive optimal schedules (see [16] for a thorough review), the schedule may be very sensitive to delays when the network is near capacity. High capacity utilization leads to more complex dispatching where small delays are created, leading to larger deviations from the train plan; this is referred to as *knock-on delay* [3, 1, 17].

Highly variable runtimes increase operational uncertainty for the railroad and for other transportation systems that depend on them. On the rail network, propagation of delay to other trains is significant [18], and there are large direct costs incurred due to additional operating time alone [19]. Delays on the rail network can also influence non-rail transportation services. For example, surface street traffic and emergency vehicles, which conflict with rail freight traffic at grade crossings [20], can be significantly delayed if a grade crossing is occupied by a train for an extended period of time. If accurate, real-time ETAs are made available, revisions to the operating plan can be implemented, and surface street transportation services can be re-routed.

1.2 Problem statement and solution approach

The main focus of the present article is the prediction problem for ETAs on US freight railroads using real-time data in an *online* setting. Compared to *offline* or *batch* algorithms, the online estimation problem requires new ETAs to be produced as new information becomes available, (i.e., as time elapses and the train progresses down the line of road). Each time the train reaches one of a number of fixed locations on the track, data is collected and a new estimated travel time to the destination is produced.

To produce the ETA estimate, a variety of routinely collected and maintained data sources available to freight railroads are used. This includes track geometry data (containing grade and curvature information, single and multi-track territory, length of sidings, etc.), historical runtimes of all trains, properties of all trains (such as length and tonnage), and crew records.

Several methodologies to produce ETAs are available, including microscopic simulation [21, 22, 23], analytical approaches [24], and data-driven techniques [25, 26]. Due to the complexity of the freight rail network (which limits the accuracy of analytical abstractions) and the difficulty to capture all delay inducing factors in a simulation based model (e.g., decisions made by human dispatchers, special cases involving priority elevation, unplanned maintenance, and weather), a data-driven approach is proposed in this article [27]. This approach is made possible through access to one of the largest and most comprehensive freight rail datasets, which is described in this article.

Location-specific characteristics are critical in any approach. For example, train length will vary in its impact on runtime depending on the lengths of sidings available on each route and the track grade. These parameters can be learned by a data-driven model trained on a historical dataset for each location, in lieu of explicitly encoding them with location-specific characteristics. Moreover, because of the increasing availability of data in this work, all parameters may be assess as they vary in time and in space.

1.3 Related work

Several lines of research are related to the problem of ETA prediction. We briefly summarize the most closely related works, and direct the interested reader to the comprehensive reviews available in the works by Bonsra and Harbolovic [25] and Gorman [28].

The majority of freight trains operate according to a schedule that is constructed in an offline manner and robust to some random unplanned disturbances [12, 3]. When extreme disturbances cause the original schedule to deteriorate, online rescheduling measures must be implemented to account for the delay and to maintain robustness to further delay [29, 14, 30]. Numerous efforts are aimed at understanding and quantifying the causes of delay that influence scheduling, rescheduling, and predictability [1, 31, 9]. Delay is typically formulated in terms of deviation from a train schedule or historical performance, but it can be extended to arrival time prediction for individual trains [25].

Several works have proposed to empirically produce delay or runtime esti-

mates using historical data for passenger rail networks. Kecman and Goverde [4] proposed an ETA prediction framework for passenger rail arrival time prediction using track occupancy data for conflict evaluation. Chapuis [32] used artificial neural networks to predict arrival times of frequent passenger trains using historical train and station delays. Compared to the proposed work, the ETAs were evaluated in the Netherlands and France, respectively, on high priority passenger traffic [33, 34], which also operates with higher punctuality compared to the freight or passenger traffic in the US [35]. Wang and Work [26] estimate passenger rail delays on the Amtrak passenger rail network in the US using vector regression techniques and only historical runtimes between passenger stations. The regression problems are formulated in both a historical and online perspective, but the feature set for prediction is limited and does not contain any data on the freight traffic, which constitutes the majority of traffic on the shared line of road in the US. Online methods presented for passenger rail, accessible because of the data stream created by station arrivals and departures, have not been fully extended to freight rail. Additionally, magnitude of delay for passenger rail is typically on the order of minutes, while delay for non-priority freight rail traffic may exceed multiple hours.

The most closely related estimation works on freight trains are the works of Gorman [28] and Bonsra and Harbolovic [25]. In Gorman [28], an econometric analysis of free-running and congestion related factors are used to identify the primary causes of delay. The data was partitioned by geographic area and priority groupings. Congestion related factors, such as meets, passes, and overtakes that occur, are found to have the largest effect on delay. Bonsra and Harbolovic [25] predict runtimes for individual freight trains in an offline setting. Prediction improvements were attained when estimated at the time of departure. The regression model used train and network parameters and a historical runtime averaging technique for evaluating model performance.

1.4 Outline and contributions

The main contribution of this article is to show how to pose the online ETA prediction problem on a rail network as a sequence of machine learning regression problems, where one regression is performed for any given origin-

destination pair. We provide practical insights by highlighting the datasets available to perform the prediction and describing some of the feature engineering required when the feature vectors change in time and space. We present a set of data features and several machine learning regression algorithms used to achieve more accurate ETAs than common statistical methods yield. Finally, the resulting models are discussed in detail with respect to their performance.

The remainder of the article is organized as follows. Section 2 presents the framework used to process and operate on the various data sources and the preliminaries for the machine learning regression. Section 3 discusses the datasets and the work that is necessary to process the data for use in the machine learning framework as constructed data features. Section 4 details the model trials that are conducted as well as the means by which to evaluate them. Section 5 describes the process for tuning and evaluating a single model, which is then extended to models across the full testing route; results are given for models using select feature combinations. The conclusion in Section 6 summarizes the progress of this research and outlines the next steps planned for investigation.

CHAPTER 2

FRAMEWORK AND PROBLEM FORMULATION

This section briefly describes the machine learning framework used for online ETA estimation and the implemented SVR algorithm [36, 37]. It reviews general machine learning terminology and parameters specific to the algorithms, both used later during analysis and discussion.

2.1 ETA Machine learning framework

The problem of predicting an estimated time of arrival for a train from an origin point to a destination point on the rail network is posed as a supervised machine learning regression problem. The goal of the regression problem is to predict the true runtime $y(i) \in \mathbb{R}^1$ of a train i given the properties of train i, the network, and other traffic on the network, which are contained in the feature vector $x(i) \in \mathbb{R}^n$. Given a dataset of m trains with true runtimes $Y \in \mathbb{R}^{m \times 1}$ and corresponding feature vectors $X = [x(1), x(2), x(3), ..., x(m)]$, where $X \in \mathbb{R}^{n \times m}$, the machine learning regression problem is to find a mapping $f \colon \mathbb{R}^n \rightarrow \mathbb{R}^1$ such that $f(x(i))$ is an accurate predictor of $y(i)$. In general, supervised machine learning regression uses a set of *training data* $\{X_{tr}, Y_{tr}\}$ (where the subscript tr is used to indicate the training data) to learn the function f, by minimizing prediction error between $f(x(i))$ and $y(i)$ over the m records in the training data.

The machine learning model f must generalize (i.e., make good predictions on data that has not been used to train the model), in order to maintain high accuracy on new data and to avoid *overfitting* the training data. To test the degree of generalization, the accuracy of the prediction is assessed on hold out *testing data* $\{X_{te}, Y_{te}\}$, which is not used to train the model.

The central difficulty of posing the online train ETA prediction problem into the framework above stems from the fact that many of the features

used for prediction change in time and in space as the train moves towards the destination. For example, the amount of traffic on the line of road will change as trains enter and leave the line of road. The number of available sidings on a route in single track territory also changes across the route and as other trains occupy or vacate sidings. If a single model is used for all origin-destination predictions in the network, it may be difficult to predict area-specific delays (e.g., due to local dispatching decisions and route characteristics) that may not occur throughout the network; results pertaining to origin-destination specificity are discussed in Section 5.2. Moreover, because some features change over time (as described above), while others may not (e.g., train priority), construction of an unbiased training dataset is a nontrivial engineering task. For example, one cannot simply create a new training data point each time a single property of a train changes (e.g., corresponding to a new feature vector) without biasing the training data, since the feature vector still corresponds to the same train trip.

2.2 Towards online, network-wide prediction

To address these difficulties, we propose to build a distinct regression model for each origin-destination pair for which predictions are required. Because the models are independent, each model can be trained using all trips that pass between the corresponding origin-destination pair by constructing features according to the state of the train and network at the time the train reaches the origin node. Localized and geography-specific performance effects may be captured in the individual models without explicitly constructing them in the feature vector. For example, longer travel times will be observed for heavy or underpowered trains in areas of high track grade. Feature construction can also vary between models (e.g., in dimensionality) since each uses a custom dataset built for the origin-destination pair.

The primary disadvantage of building a model for each origin-destination pair in the network is the number of models required. In a rail network with k nodes, in principle k^2 predictors are required. In contrast to road networks where the number of nodes and the number of viable paths between any two node pairs may be large, rail networks have fewer nodes and less path redundancy. In practice, few locations are relevant destination points

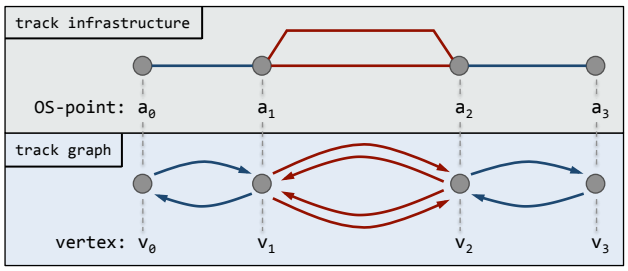

Figure 2.1: Graph vertices align with OS-points and each track segment between them is represented by a graph edge in each direction. Double track areas and sidings, therefore, are represented by four graph edges in order to enumerate all unique routes across the network.

from a given origin because a single route (excluding small deviations for sidings) is typically used to connect two points on the network. Therefore, the number of relevant origin-destination paths for which predictions are required is tractable. In the area of study in this work, there exist 35 points that can serve as origin points; there are only four practical destination points from each origin point, which results in at most 148 predictors.

In order to map spatiotemporal train data to the network topology, infrastructure data can be reconstructed into a directed graph format, $G = (V, E)$ where V is a set of vertices and E is a set of directed edges. Vertices are points where the track merges and splits (e.g., endpoints of sidings). Data on passing trains is recorded at *OS-points*, which are fixed locations $v \subset V$. Directed edges represent track segments across which trains travel between OS-points and a direction of travel. This alignment between track infrastructure and the constructed graph is shown in Figure 2.1. OS-points are denoted a_0, a_1, a_2, a_3 with corresponding graph vertices v_0, v_1, v_2, v_3. Pairs of directed edges representing each delineated track segment allow distinct runtimes and feature values in each direction, which is necessary when properties such as grade are considered [25]. In this formulation, all trains can be routed on the graph across their unique path considering track usage (e.g., siding track versus main line track). Data can be gathered on the behavior of trains for each directed edge with respect to speed and other train attributes. Also, features that consider estimates of the positions of multiple trains and track topology can be mined from this data.

2.3 Summary of predictors

We briefly describe the primary regression algorithm explored in this work, as well as the naive benchmark used to assess the performance of the regression approach.

2.3.1 Regression via support vector machines

The regression problem of predicting ETAs from a vector of features is solved with a *support vector regression* (SVR) machine, first introduced in [38]. Support vector regression is a popular machine learning algorithm grounded in statistical learning theory and for which training is efficient due to the convexity of the training problem. In the training step, the optimal regression parameters are selected to minimize prediction errors larger than a threshold ε, while penalizing complex models through a norm on the feature weights. Precisely, the loss function is an ε-insensitive loss $|\cdot|_\varepsilon$, constructed as [39]:

$$|\xi|_\varepsilon = \begin{cases} 0 & \text{if } |\xi| \leq \varepsilon \\ |\xi| - \varepsilon & \text{otherwise,} \end{cases} \tag{2.1}$$

where ξ is a prediction residual. The SVR formulation also provides for extension to nonlinear regression via kernel functions. Additionally, the model parameters in linear SVR are straightforward to interpret, which can be invaluable in the application of the algorithm. Other applicable algorithms include linear ridge regression [40], elastic net regression [41], and kernel ridge regression, to name a few.

The training step in SVR involves the following convex optimization problem:

$$\begin{aligned} \underset{w,b,\xi,\xi^*}{\text{minimize}} \quad & \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{m}(\xi(i) + \xi^*(i)) \\ \text{subject to} \quad & y(i) - w^T x(i) - b \leq \varepsilon + \xi(i), \\ & w^T x(i) + b - y(i) \leq \varepsilon + \xi^*(i), \\ & \xi(i),\ \xi^*(i) \geq 0. \end{aligned} \tag{2.2}$$

Feature weights are denoted $w \in \mathbb{R}^n$ and the two norm of these weights is included in the minimization in order to penalize an overly complex model that is overfitted to training data. This characteristic is referred to as model

flatness. Residual prediction errors are denoted by $\xi^{(*)} \in \mathbb{R}^m$ and are non-zero for predictions outside the margin ε. The problem is subject to the constraints of the ε-insensitive loss function, where $w^T x(i) + b = f(x(i))$. The total ε-insensitive loss (i.e., accuracy of model fit) is balanced against model flatness by a scalar factor C. The value $b \in \mathbb{R}^1$ is the regression intercept.

The optimization of (2.2) can be solved via the Lagrange dual problem, which introduces the Lagrange multipliers $\alpha(k) \in \mathbb{R}^1$ and $\alpha^*(k)$ learned from the training data in the dual problem. These dual variables provide for solving the feature weights by leveraging the relationship $w = \sum_{i=1}^{m}(\alpha(i) - \alpha^*(i))x(i)$. This results in a predictor of the form:

$$f(x) = \sum_{i=1}^{m}(\alpha(i) - \alpha^*(i))x(i)^T x + b \tag{2.3}$$

The resulting $f(x) = w^T x + b$ minimizes (2.2) by choosing the best w via the dual variables; see [36] for a comprehensive description.

When the ETAs in the training data are not linearly related to the features, an alternative strategy is to transform the training data into a much higher dimensional feature vector denoted by $\Phi(x)$, where $\Phi\colon \mathbb{R}^n \to \mathbb{R}^N$ with $N >> n$, which can then be used for regression. The new predictor becomes:

$$f(x) = \sum_{k=1}^{m}(\alpha(k) - \alpha^*(k))\Phi\left(x(k)\right)^T \Phi(x) + b. \tag{2.4}$$

Interestingly, it is not necessary to explicitly define the mapping to the high dimensional space, since only the inner product $\Phi^T\Phi$ is needed in the regression function. The inner product can instead be defined through a kernel function $K(x(k), x(j)) = \Phi\left(x(k)\right)^T \Phi(x(j))$. The use of the kernel function directly in (2.4) is known as the *kernel trick* [42] in machine learning. In the present work, we adopt the *radial basis function* (RBF) kernel [43] of the form:

$$K(x(k), x(j)) = \exp\left(-\frac{1}{2\sigma^2}\left\|x(k) - x(j)\right\|_2^2\right), \tag{2.5}$$

where σ is a parameter controlling the decay rate of the kernel, effectively limiting the influence that any single observation may have on the trained model.

CHAPTER 3

FEATURE CONSTRUCTION AND DATA CLEANING STEPS

This section discusses the process of combining and mining datasets to be used in feature construction. The data used in this work is described first, before describing the features that are calculated from the datasets which are subsequently used to train the machine learning ETA algorithm. Due to the proprietary nature of the data, some descriptions are reported in relative terms.

3.1 Description of raw data

This work uses a series of datasets describing the rail network and operations from December 1, 2014 through January 31, 2017 inclusive. It consists of freight train movement, train car operations, crew, and locomotive data in the CSX Transportation network, extracted from dispatching and signaling data.

The movement data consists of records generated at OS-points between terminals. The data includes the track on which the train was reported and the time at which the train triggered the OS-point. This dataset also contains information about track mileage covered, direction of travel, and the next location at which work on the train will occur.

The train car operations data details the actions performed on the train once it enters the terminal from the line of road. This includes the switching operations (i.e., picking up and setting out rail cars) that are referred to as train *work*, inspections, refueling, and crew changes that are scheduled to occur at intermediate destinations/terminals and may incur delay in getting track space in the terminal. The planned work schedule, as well as adherence to the schedule, is reported in this data. Changes in physical train characteristics (e.g., total number of cars, length, tonnage) are inferred based on

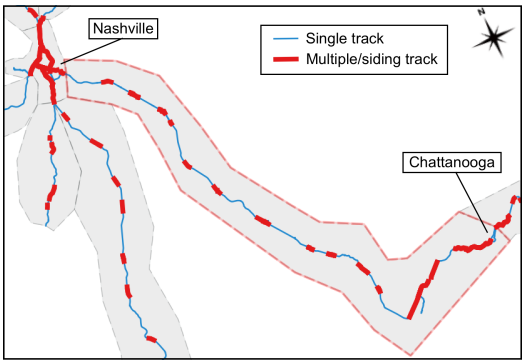

Figure 3.1: GIS network view depicting a portion of the Nashville division, with multi-track segments shown in bold red lines and single-track segments in thin blue lines. The primary study route is bounded with the dashed line.

the work reported on the train.

Crew data contains information about the crew assigned to the train, the originating location, the time at which they were called on duty, and the time at which the crew must legally go off duty (i.e., 12 hours after going on duty). The time between a crew going on duty and the departure of the train is non-negligible and is referred to as *on duty time to departure* (ODTOD). Crew information is important because the maximum crew on-duty requirement must always be satisfied, even at the large expense of stopping a train and transporting a replacement crew to finish the trip.

Locomotive assignment data indicates the equipment and total locomotive power available on each train, which can be important for predicting delays in regions with large grades.

This work also uses GIS data describing the physical infrastructure of the network, which includes individual tracks, switches, mileposts, and terminals. All locations referenced in the movement, work, crew, and locomotive data map to physical infrastructure locations, such as track mileposts and control points. Reconciling these GIS data sources is necessary to gather data on the number of tracks and siding locations and lengths on each route and build the network graph described in Section 2.2. Figure 3.1 depicts this data, along with the distinction between single track sections (shown by the thin blue lines) and multiple track sections or sidings (shown by the bold red lines). The study area is bounded by the dashed line.

3.2 Data cleaning and standardization tasks

A variety of data cleaning and data transformation tasks are necessary to organize the input for any prediction algorithm. With over 150,000 trips in the division in a two year period, the decision was made to neglect trips with data completeness issues or data errors instead of devising a scheme to impute missing or erroneous values; this resulted in the discarding of approximately 10% of trains. Errors consist of fields that contain missing data, or fields that contain illogical values. Examples include non-physical train lengths, or an arrival time prior to the train departure time.

The GIS data is examined to ensure proper connectivity and accuracy before being transformed into the network graph. Common errors encountered include duplicate geometries, disconnected track components, and minor mislabeling of infrastructure components. Many errors are automatically identified and resolved, while some errors require manual correction.

The detection and resolution methods for each of these data fields are summarized in Table 3.1. Each was implemented at the time of data mining and feature construction, so that an origin-destination dataset is clean at the time of model training.

3.3 Handling of recrewed trains

In the process of early prediction efforts and data exploration efforts, a dominant source of runtime variability was discovered. Specifically, it was found that *recrewed* trains (i.e., a train that did not reach its destination before the crew reached its maximum on-duty time and needed a relief crew) define the dominant source of runtime variability on the study route.

To further investigate the impact of recrews on train variability, all trains were ex post facto labeled as either recrewed or non-recrewed. Less than 10% of the trains on the route were recrewed. The two classes (recrewed and non-recrewed trains) were separated and descriptive statistics were calculated for each class at each of the 35 OS-points, which are spread across the route depicted in Figure 3.1. The standard deviation of runtimes was used to quantify the runtime variability of trains in each class as well as the variability of all trains in the dataset (not separated on recrew). As shown in Figure 3.2,

Table 3.1: Data cleaning and standardization steps including detection and fix.

Data field	**Data error**	**Data correction**
Train arrival time	arrival time $\leq$ departure time	discard train
Train length	length ≤ 0	discard train
Train tonnage	tonnage ≤ 0	discard train
Train horsepower	horsepower ≤ 0	discard train
Train direction	direction switches on route (indicates train work)	discard train
Crew assignment	no crew assigned to train	discard train
Crew on duty time	on duty time $\geq$ train departure time	discard train
Track segments	polylines connect spatially, but endpoint ID mismatch	detect spatial connection, resolve endpoint ID mismatch
Duplicate GIS elements	identical geometry encoded strings	check connectivity of each, keep most connected

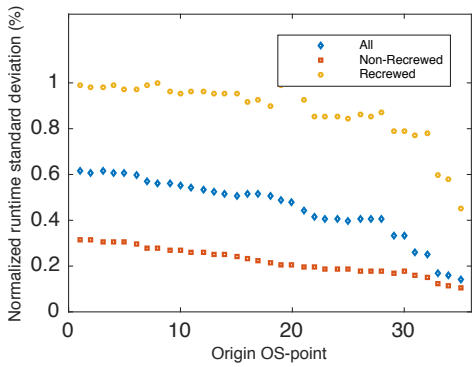

Figure 3.2: Comparison of variability in runtime, between recrewed trains, non-recrewed trains, and all trains, for each origin OS point. Variance is normalized by the largest variance OS-point, OS-point #21. OS-point IDs increase from Nashville (1) to Chattanooga (42).

the runtime variability of the recrewed trains is several times larger than that of the non-recrewed trains across all OS-points; runtime variability is expressed as a relative value to protect proprietary operational properties in the data. Despite recrewed trains representing less than 10% of the trips, they represent 53% of the variability within the dataset of all trains, when averaged across the full route.

Recrewed trains introduce high variability in runtime and their runtimes are not predictable by features inside the scope of the train and network state features (e.g., location and status of potential relief crews is a significant factor and is not in our dataset). It is likely, however, that the circumstances leading to a recrew will enable its preemptive classification and could be captured with the available data. For the scope of this project, recrewed trains were historically identified and removed from the training data as part of data cleaning and standardization steps.

3.4 List of calculated features

This section lists and discusses the features that are generated from the raw data in Section 3.1 and used to train machine learning algorithms in ETA prediction. A summary of the implemented scalar features appears in Table 3.2. These features include six train characteristics, two features that capture the state of the crew on each train, and multiple scalar features quantifying the

network characteristics and traffic. Numerous series of features describe the traffic state of the network in the vicinity of prediction. These all depend on segmentation of the track between OS-points. The network traffic state is described for these segments in terms of occupancy, direction, and priority; these are summarized in Table 3.3. All of the features chosen for exploration were based on extensive discussions with operations research personnel from CSX Transportation.

The priority of a train is determined by its cargo (e.g., bulk, merchandise, automotive, intermodal), and its type of service (e.g., local, yard, road). These combinations are categorized and ranked in different levels of granularity. At the highest resolution, all trains are placed into one of twenty priority classes. The relative priority ranking is understood to be non-linear based on its construction. The high resolution ranking was aggregated to a medium-resolution ranking using five priority classes and a low-resolution ranking of three priority classes. For example, scheduled merchandise trains have significantly higher priority than bulk/unit trains (e.g., loaded coal train), but in medium- and low-resolution classifications, the two types will get the same priority designation.

The physical train characteristics such as train length and train tonnage were calculated by examining the work data which contains the train dimensions after the most recent work was completed. Similarly, the most recent crew change can be identified in the crew data, which is used to calculate the maximum time the current crew may continue to operate the train, called *crew time remaining*. The practical importance of this feature is in terms of the difference between crew time remaining and expected train runtime (constant), called *slack time*. If the crew time remaining is less than the typical train runtime, then a train may be expedited in order to avoid a recrew. In terms of feature construction, crew time remaining and slack time reduce to the same values under min-max normalization applied to each element in the feature vector used for the model input X.

A total of six scalar quantifying measures of traffic are constructed, each based on the line of road between the current location of a train and the destination. First, within the remaining line of road, all other trains including local trains are counted, and then categorized based on the direction of travel for each (e.g., same direction, subscript ω, or opposite direction, subscript *psi*, relative to the train being predicted). A second parameter categorizes

directional traffic counts (two counts) by their priority, as higher (subscript α) or lower/equal priority (subscript β) relative to the train being predicted, resulting in another four features.

The exact traffic state of the origin-destination route and surrounding area is described further using numerous feature series, the components of which correspond geographically to track segments. The segments refer to the connections between OS-points, and span the origin-destination route and distances around the origin and destination limited to 50% of the origin-destination route length. The dimension of each feature series is equal to the number of track segments that are considered. In our study area of approximately 140 miles, the 35 OS-points delineate the segments on the origin-destination route. OS-points within 70 miles of the origin and 70 miles of the destination not included on the origin-destination route delineate the track segments around the origin and around the destination. These series features are enumerated and explained in Table 3.3. Segment occupancy is a series of binary features, each of which is non-zero when another train is present in a track segment at the time that a prediction is made for a train at the origin point. Likewise, trains that are present on these segments can be described with respect to their relevant properties, namely direction of travel and priority. This process of describing the traffic state via network segments and traffic properties is illustrated in Figure 3.3. Predictions are made at the origin OS-point a_0 and OS-points delineating segments are labeled a_0 through a_l (for origin-destination route, only). An example traffic scenario for the moment at which a prediction is made for train Z_0 at the origin is shown with trains Z_1, Z_2, Z_3. The relevant features (i.e., direction and priority) of each train are listed, and are mapped to the track segments corresponding to the location of each train.

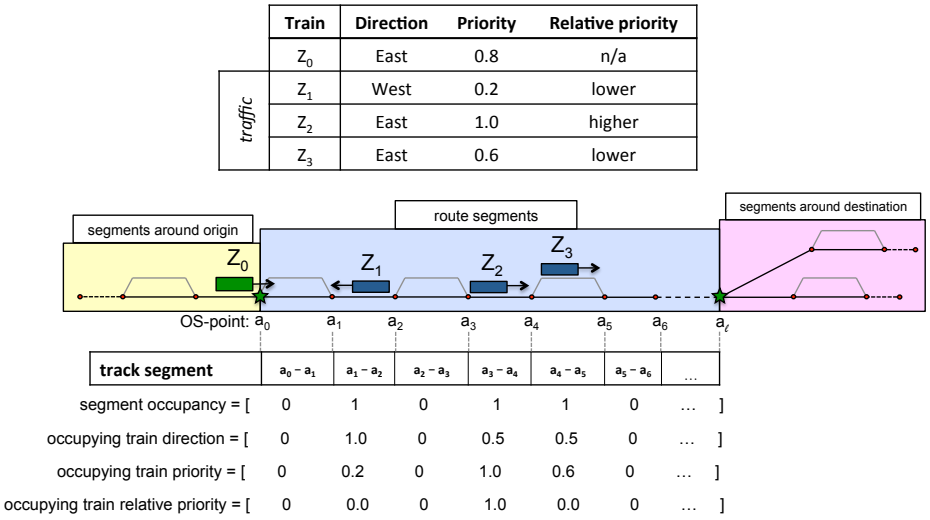

Figure 3.3: Segment-wise series features are calculated for the area around an origin point, on the origin-destination route, and around the destination. Each is segmented by OS-points, a_0 through a_l on the origin-destination route in this case. The occupancy series feature is constructed, followed by series based on properties of the occupying train, if present.

Table 3.2: Summary of implemented scalar features.

Feature	**Notation**	**Description**
Train length	λ_i	The total length of locomotives and cars of train i.
Train tonnage	μ_i	The total mass of locomotives and cars.
Train horsepower per ton	η_i	Total horsepower of locomotives divided by train tonnage, μ_i.
Train priority (high-resolution)	$\rho_{20,i}$	Priority ranking on a 1-20 scale.
Train priority (medium-resolution)	$\rho_{5,i}$	Five priority classes are constructed by aggregating the high-resolution priority ranking.
Train priority (low-resolution)	$\rho_{3,i}$	Three priority classes are constructed by aggregating the high-resolution priority ranking.
Crew time remaining	γ_i	Amount of time remaining that the current train crew can legally work.
On duty time to departure	θ_i	Time between crew on duty time and train departure.
Full traffic count	τ_i	Count of trains on the remaining line of road.
Directional traffic count	$\tau_{\omega,i}, \tau_{\psi,i}$	Count of trains on the remaining line of road, divided by direction of travel (i.e., in the same direction, ω, or in the opposite direction of travel, ψ).
Prioritized directional traffic count	$\tau_{\omega,\alpha,i}, \tau_{\omega,\beta,i}, \tau_{\psi,\alpha,i}, \tau_{\psi,\beta,i}$	Count of train on the remaining line of road, divided by both direction of travel and priority relative to that of the train being predicted (i.e., lower or equal priority, β, or higher priority, α).
Available sidings	π_i	Count of sidings on route with length greater than that of train i.

Table 3.3: List of calculated and implemented series features, which correspond in dimension to the segmentation of the remaining route, and details for each.

Feature	**Notation**	**Description**
Track segment occupancy	$O_i = [O_{1,i}, ..., O_{l,i}]$	Vector of elements denoting whether a segment (beginning at segment 1 for l additional segments to the destination) is occupied by another train, non-zero when occupied.
Occupying train direction	$D_i = [D_{1,i}, ..., D_{l,i}]$	Denotes the direction (same or opposite) of a train occupying a track segment (beginning at the first segment (segment 0) for l additional segments); zero if no train.
Occupying train priority	$P_i = [P_{1,i}, ..., P_{l,i}]$	Assigns high-resolution train priority values as described above to a train occupying a track segment. Higher value for high priority train; zero if no train.
Relative priority of occupying train	$R_i = [R_{1,i}, ..., R_{l,i}]$	Non-zero when a train occupying a track segment has higher priority than the train for which the ETA is being predicted. Reflects likelihood of meet/pass delay.
Track segment occupancy around origin point	$G_i = [G_{-1,i}, ..., G_{-h,i}]$	Indicates track segment occupancy for segments -1 through $-h$ around the origin point, but not included in the primary route (segments 0 to l); captures trains that may enter the primary route and pass/overtake
Track segment occupancy around destination point	$E_i = [E_{l+1,i}, ..., E_{l+p,i}]$	Indicates track segment occupancy for segments $l+1$ through $l+p$ around the destination point, but not included in the primary route; captures trains that may enter the primary route and conflict during the trip

CHAPTER 4

MODEL IMPLEMENTATION AND EVALUATION

This section describes a set of model experiments and a metric to assess the machine learning methods described above. The feature sets used in the models are composed of the features described previously in Section 3.4.

4.1 Description of models

Numerical experiments were performed with concentration on a single route, shown by the dashed area in Figure 3.1, in the Nashville division of the CSX Transportation network which contains a mix of single and double track segments, highly heterogeneous traffic, and high volume relative to capacity. The route represents one of the most challenging segments on which to estimate ETAs within the CSX Transportation network. Without loss of generality of the methods, the present analysis is restricted to common train types with sufficient trips in the two year dataset and includes the automotive, merchandise, and intermodal trains. These train types have differing priorities, and consequently have distinct behaviors in meet/pass movements and when delays occur. The dataset for trains running the full study route in the correct direction of travel initially contains over 10,000 trips. When the dataset is filtered by train type, recrewed trains are removed, local trains and trains with intermediate work are eliminated, and data errors and incomplete records are removed, there are still approximately 4,200 trips.

The selected route is composed of 35 points along the 140 mile route for which an ETA to the destination must be produced. For each of the 35 ETA problems, a total of five models are implemented and compared. The models include the baseline median predictor algorithm as well as four SVR-based algorithms. Many combinations of algorithm type and feature set were explored, and the presented models are representative of the model type and

performance. For example, the various priority features were each evaluated for predictive performance by performing single-feature model trials and $\rho_{5,i}$ was found to be the most informative.

The exact model configurations are as follows:

- Model 0: baseline median predictor where $f(x(i)) = median y(i) \mid y(i) \in Y_{tr}$
- Model 1: linear SVR with all scalar features (length, tonnage, hp/ton, priority, crew time, ODTOD, traffic counts, and sidings fit); the feature vector is constructed as: $x(i) = [\lambda_i, \mu_i, \eta_i, \rho_{5,i}, \gamma_i, \theta_i, \tau_i, \tau_{\omega,i}, \tau_{\psi,i}, \tau_{\omega,\alpha,i}, \tau_{\omega,\beta,i}, \tau_{\psi,\alpha,i}, \tau_{\psi,\beta,i}, \pi_i,]$, where $x(i) \in \mathbb{R}^{14}$.
- Model 2: linear SVR with all scalar features plus track segment occupancy $x(i) = [\lambda_i, \mu_i, \eta_i, \rho_{5,i}, \gamma_i, \theta_i, \tau_i, \tau_{\omega,i}, \tau_{\psi,i}, \tau_{\omega,\alpha,i}, \tau_{\omega,\beta,i}, \tau_{\psi,\alpha,i}, \tau_{\psi,\beta,i}, \pi_i, [O_{1,i}, ..., O_{l,i}]]$, where $x(i) \in \mathbb{R}^{14+l}$.
- Model 3: linear SVR with all scalar features and all series features $x(i) = [\lambda_i, \mu_i, \eta_i, \rho_{5,i}, \gamma_i, \theta_i, \tau_i, \tau_{\omega,i}, \tau_{\psi,i}, \tau_{\omega,\alpha,i}, \tau_{\omega,\beta,i}, \tau_{\psi,\alpha,i}, \tau_{\psi,\beta,i}, \pi_i, [O_{1,i}, ..., O_{l,i}], [D_{1,i}, ..., D_{l,i}], [Q_{1,i}, ..., Q_{l,i}], [R_{1,i}, ..., R_{l,i}], [G_{-1,i}, ..., G_{-h,i}], [E_{l+1,i}, ..., E_{l+f,i}]]$, where $x(i) \in \mathbb{R}^{14+4l+h+p}$.
- Model 4: RBF kernel SVR with all scalar features and all series features $x(i) = [\lambda_i, \mu_i, \eta_i, \rho_{5,i}, \gamma_i, \theta_i, \tau_i, \tau_{\omega,i}, \tau_{\psi,i}, \tau_{\omega,\alpha,i}, \tau_{\omega,\beta,i}, \tau_{\psi,\alpha,i}, \tau_{\psi,\beta,i}, \pi_i, [O_{1,i}, ..., O_{l,i}], [D_{1,i}, ..., D_{l,i}], [Q_{1,i}, ..., Q_{l,i}], [R_{1,i}, ..., R_{l,i}], [G_{-1,i}, ..., G_{-h,i}], [E_{l+1,i}, ..., E_{l+f,i}]]$, where $x(i) \in \mathbb{R}^{14+4l+h+p}$.

4.2 Model evaluation

The error metric used to evaluate a given model is *mean absolute error* (MAE), defined as:

$$MAE = \frac{1}{p} \sum_{i=1}^{p} |f(x(i)) - y(i)|, \tag{4.1}$$

where $f(x(i))$ and $y(i)$ correspond to the predicted runtime and true runtime of train i, respectively, and p denotes the number of records in the testing dataset. It follows that low MAE *scores* are better than high scores. Performance of each model is compared to that of the historical median predictor,

Model 0. The improvement for each model is given as the reduction in the MAE relative to the historical median predictor.

The machine learning algorithms and benchmark predictors are implemented in Python and leverage the scikit-learn package [44]. The data processing steps are completed once for each origin-destination pair and the feature set is stored in a database. Model trials are performed by loading the feature set, selecting the desired data, normalizing features, and training the model and testing the performance via cross-validation. Data processing and model testing both occur on a dedicated, modern workstation computer with 4 GHz quad-core processor, 64 GB DDR4 RAM, and NVMe solid state drives. Building the feature set is the most time consuming step in the process, due primarily to the size of the raw data. This process can take up to 15 minutes per origin-destination pair but is only required once, and adding more features and data does not require reprocessing of previous data. Model training is accomplished in 5.0 to 15.0 seconds, with prediction on all test data taking no more than 3.0 seconds.

CHAPTER 5

RESULTS

This section first presents the process for choosing hyper-parameters C and ε in (2.2) for a single origin-destination model with scalar features. We then analyze the series of models trained with scalar features on the full 35-OS-point route and the differences between them, specifically with respect to feature weights. Performance results are then shown for each model in Section 4.1 across the route, as well as the impact of the nonlinear kernel.

5.1 Choosing hyper parameters for a single model

For each origin-destination model, the SVR parameters, C and ε are chosen to minimize MAE on testing data. The training and testing process is performed using a dataset containing approximately 4200 trips using a 5-fold cross validation with an 80/20 training/testing data split. The parameter space is explored using a grid search that explores all combinations of parameters within the bounds of each. The trained model must be checked for suitability such that it generalizes well to testing data. This check is done using validation curves for C and ε and a learning curve for the amount of data used to train the model.

The interpretation of these parameters as well as analysis of training and testing behavior kept the search space limited. The ε parameter is directly related to residual values between $f(x(i))$ and $y(i)$ and, therefore, can be limited to a search space proportional to the normalized spread of the true outputs $y(i)$. The C parameter penalizes the model training error, summed across all observations $x(i)$, relative to the model flatness, given by the two norm of feature weights. We normalize C such that it is scaled by the number of features and inversely scaled by the number of observations in the training dataset. This maintains the impact of the parameter across models with

different dimensionalities.

A validation curve explores training and testing scores across a range of a model parameter, with other parameters fixed. Using a fixed C value of 0.1, the validation curve for the ε parameter in Figure 5.1 shows relatively little effect of ε value on training and testing scores. Within the acceptable parameter range, high values nor low values make an appreciable effect on testing MAE. Approaching $\varepsilon = 0.0$, the ε-insensitive loss disappears and the algorithm converges to linear regression. When ε is set too high, more emphasis is placed on the number of observations that lie within $\pm\varepsilon$ and less is placed on minimization of prediction error. In comparison, the value of C has significantly higher impact on model score. The validation curve is also shown in Figure 5.1 with ε fixed at 0.2 and plotted with common normalized MAE score for comparison. Small values of C emphasize model flatness, but result in low training and testing scores because model complexity is low. Large values of C achieve low training MAE but generalize poorly to the testing data because of overfitting.

Though validation curves show single parameter sensitivity, the optimal parameters are chosen simultaneously by evaluating the model on the grid space of all parameter combinations ($C \in [10^{-5}, 10^{4}], \varepsilon \in [0.0, 1.0]$. For origin-destination model at OS-point #1, the optimal values that minimize MAE are found to be $C = 0.75$ and $\varepsilon = 0.4$; the difference in training and testing scores is less than 3% of training error, which is clearly avoiding overfitting.

The learning curve for a model shows the convergence of training and testing performance by increasing the amount of data available to build the model. The curve is analyzed after hyper-parameters have been chosen. The learning curve shows the MAE score against the number of training examples available to the model. With smaller amounts of data available, training scores will be improved, but at the expense of the model generalizing poorly to testing data. The learning curve in Figure 5.2 indicates that the model is trained on a sufficient amount of data because the curves converge before the full training dataset is used. The mean training and testing scores are denoted by the lines with the shaded areas showing one standard deviation in each direction between cross-validation folds. The model training and testing occur at 10%, 32.5%, 55%, 77.5% and 100% of available data. The residual disparity in training and testing scores begins at over 40% of training score

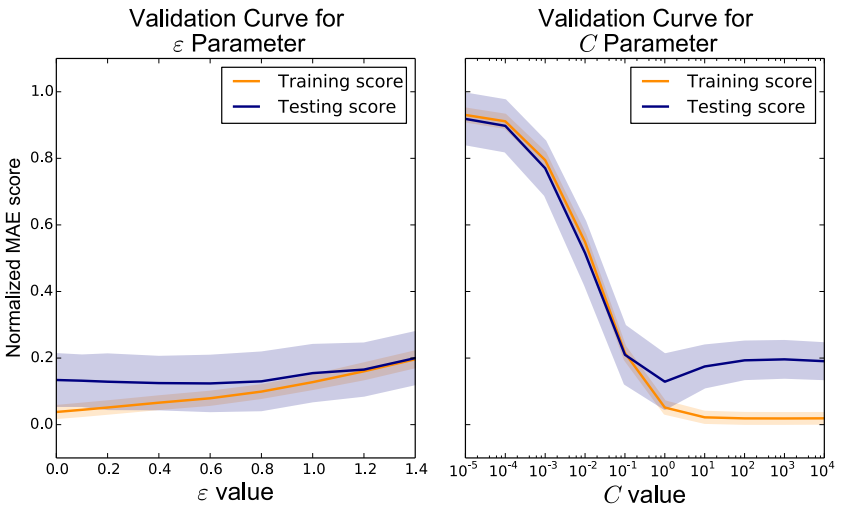

Figure 5.1: The validation curves for the C and ε parameters on a single origin-destination model are plotted with a common MAE score axis, which is min-max normalized across both parameters. The sensitivity of the model to ε is relatively low compared to that of C.

with only 10% of available data and decreases to less than 4% with 100% of available data.

Testing performance of the model can be further assessed in the distribution of errors around the true values, shown in Figure 5.3. Perfect prediction would align points on the line $f(x) = y$, with true output equal to predicted output. The predictions made by the model at OS-point #1 have smaller range than that of the true output values, meaning the model made more conservative estimates for the more extreme points. It is possible that factors causing extreme output values are occur rarely (e.g., special priority elevation or maintenance on the route) and make the relationship difficult to capture from the data.

For any SVR model with a linear kernel, the feature weights can be interpreted meaningfully in both magnitude and sign. The feature weights were recorded at all cross validation folds and for each origin-destination model to assess the relative impact of each. The feature weights are normalized by absolute sum within each model, such that $\sum_{j=1}^{n} |w_j| = 1$, to allow comparison between models. They are reported in absolute terms for ranking in terms of absolute impact. The results for the origin-destination model at OS-point #1 (the beginning of the route in Nashville) are shown in Table 5.1. For this model, the effect of train priority is the dominant feature; this is supported

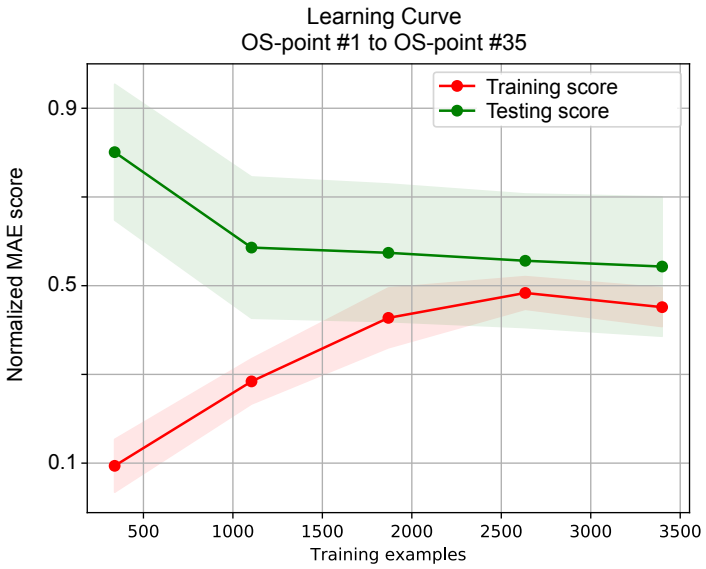

Figure 5.2: The learning curve for a single origin-destination model shows convergence of the training and testing error scores given increasing availability of observations in the full dataset.

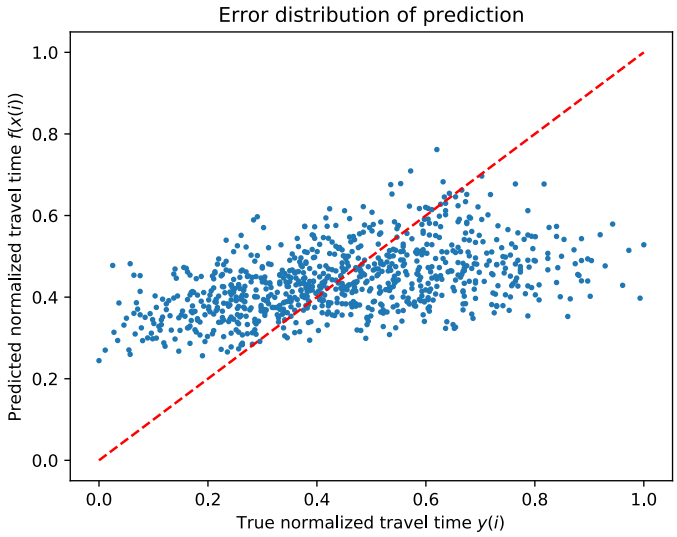

Figure 5.3: Comparison of predicted outputs $f(x(i))$ to true output values $y(i)$. Both are normalized runtimes with the origin-destination median runtime subtracted.

by the statistically distinct runtimes between priority classes at this distance from the destination. Crew time remaining has a large impact because it can affect the runtime of a train at this distance if the train experienced significant delay leaving its last terminal. Tonnage also play a large role due to the lower overall performance of the train during acceleration and deceleration. Features with particularly low impact include traffic counts separated by direction and priority, which is an overly simplistic view of the traffic state on long routes. Horsepower per ton also has less of an impact because few trains are under powered in an area such as this with significant terrain.

5.2 Model training across route

The hyper-parameter selection process and model evaluation was replicated for origin-destination predictions made at each of the 35 OS-points on the full route. Because the C parameter is normalized by the training data size and feature dimension, and the ε parameter demonstrated little sensitivity, the optimal set of hyper-parameters found by the selection process varied little across the route.

The main finding is that reported feature weights show significant vari-

Table 5.1: Average feature weights for 5-fold cross validation on origin-destination prediction at OS-point #1 (Nashville). Exact model output weights are normalized by absolute sum.

Feature	**Average weight**
Priority, P_5	0.346
Crew time remaining, C	0.137
Tonnage, M	0.110
Traffic opposite direction lower/equal priority, $T_{U,L}$	0.089
Available sidings, S	0.067
Total traffic, T	0.055
Traffic opposite direction, T_U	0.050
Length, L	0.047
Traffic same direction, T_W	0.040
Horsepower per ton, Q	0.019
Traffic opposite direction higher priority, $T_{U,H}$	0.011
Traffic same direction higher priority$T_{W,H}$	0.011
Traffic same direction lower/equal priority$T_{W,L}$	0.011
On duty time to departure, θ	0.008

ability across the route. This supports the idea that dispatching techniques have fundamental differences based on relative location of the train to a terminal point. All scalar feature weights are shown at each prediction point in Figure 5.4. The means of each feature at each location are represented by the lines and min-max ranges for each are shown by the shaded area around the lines. In prediction of the full route, at OS-point #1, the factor most heavily impacting train runtime is priority. Other factors certainly play into the dispatching decisions made for the route, but do not appear to have strong relationships far from the destination. Closer to the destination, traffic counts and train tonnage are driving factors due to decisions around the yard and natural choke point in the network. The changing importance of train characteristics supports the domain understanding that many factors contribute to train ETA and the impact of these factors is not constant. Along the route, feature weight experience some sharp changes, but some of these can be explained by certain characteristics of the route. For example, the sharp dip in the weight of priority and crew time remaining, along with the sharp increase in weight on traffic in the same direction, is located on the most significant hill on the route. At this location, a separate helper locomotive attaches to and assists trains in climbing the hill. Availability of this locomotive is a driving factor in runtime from this location and its presence is captured indirectly in the feature set without being explicitly defined. This separation of origin-destination models is, after all, intended to encode these geographic features in the data via specificity.

This specificity serves as a justification for the separation of origin-destination predictions to distinct models, as opposed to a unified model predicting ETAs for all origins and destinations. Feature weights may be learned optimally for a single model, instead of forcing feature weighting conditioned upon location information. It is possible that some features may change weight predictably relative to the distance from the destination point such that a distance normalization in feature construction could account for weight change. However, the results of feature weights across the route show that many of the trends are inconsistent.

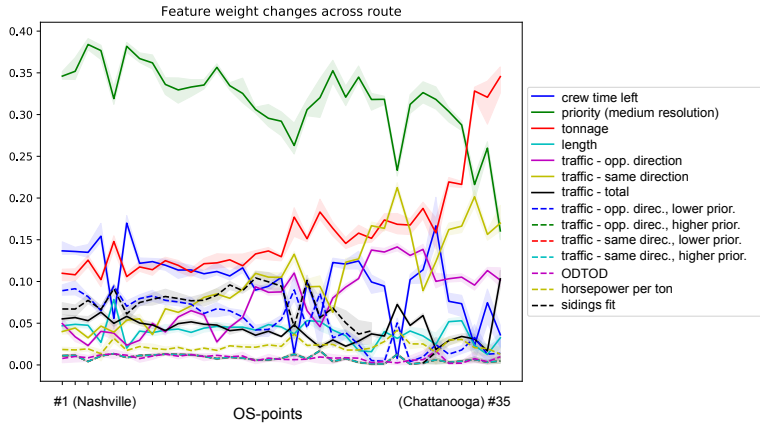

Figure 5.4: Feature weight change of scalar features in origin-destination SVR models across the route, Nashville (1) to Chattanooga (35).

5.3 Performance comparison of SVR models

Performance of each model detailed in Section 4.1 is evaluated across all OS-points on the route. The four non-baseline models demonstrate increasingly levels of model complexity based on the features included in each, but are not a comprehensive list of all feature combinations that were tested.

The model results are shown in Figure 5.5 in terms of relative reduction in MAE over the historical median predictor (Model 0). A features set with only scalar features (i.e., Model 1), as explored in the choice of hyper-parameters and examination of feature weights, represents the largest performance gain for every origin-destination predictor. Inclusion of the segment-wise occupancy feature series (Model 2) and inclusion of all segment-wise features series (Model 3) each attain small MAE improvements in addition to improvements gained by the scalar features. The RBF kernel, however, does not provide any substantive performance gain over the fully featured linear model. The γ parameter in the RBF kernel was chosen by exploring the range $\gamma \in [10^{-}4, 10^{3}]$ in a grid search alongside the ε and C parameters.

Based on the relative performance improvements of the SVR models over the baseline, it is evident that the addition of the segment-wise network traffic state features (Models 2 and 3) show clear advantages by providing

Table 5.2: Comparison of SVR model performance to baseline predictor, averaged across the 35 OS-points on the full route, for each model.

Predictor	**Mean % improvement**	**Maximum %**	**Minimum %**
Model 0 (Baseline)	0.0%	0.0%	0.0%
Model 1	9.4%	14.2%	4.0%
Model 2	12.2%	19.9%	4.6%
Model 3	14.0%	21.6%	6.2%
Model 4	14.3%	21.8%	7.0%

high-resolution information compared to the simple counts of network traffic (Model 1). The larger gains are achieved by the segment-wise occupancy feature series, but the additional information on the direction and priority of the traffic improve the model performance by better informing on the likely meets and passes that will occur. For instance, the mere presence of another train on the route will be somewhat likely to increase runtime, but if this train is traveling in the direction opposing trains on the origin-destination route and has high priority then it may be highly likely to increase runtime.

Model performance varies somewhat across the origin OS-points due to the distribution of route delay. The spacing of sidings and the likelihood of each to be used varies due to their length and topological characteristics of the route. Overall, the relative performance of the models to each other is consistent across the route. Prediction performance decreases closer to the destination. We expect this is due to the more unpredictable nature of operations close to rail yards. The factors that affect the exact arrival of a train when it gets close are not necessarily present in the data (e.g., ability of the yard to accept more trains, availability of the next train crew)

These route results are summarized in Table 5.2 in terms of mean, maximum, and minimum percent improvement over the baseline. The minimum improvement values are consistently observed for models close to the destination point and the maximum improvement is observed at the beginning of the route.

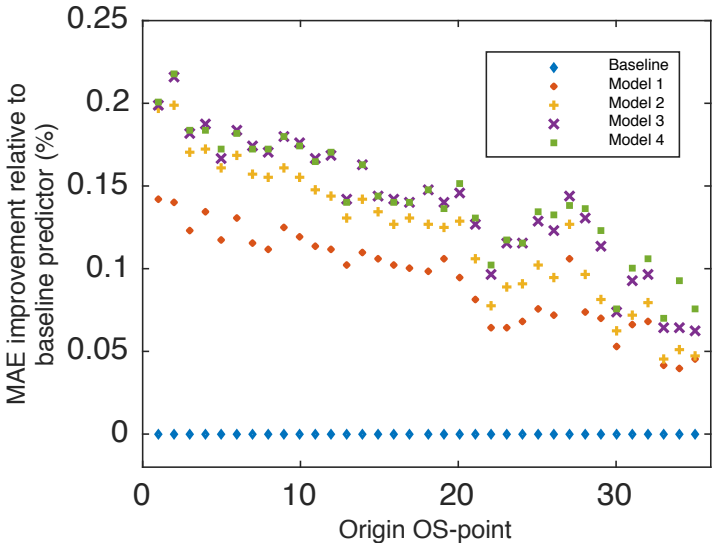

Figure 5.5: Improvement in MAE over baseline historical median predictor for each model at all OS-points between Nashville (1) and Chattanooga (35).

CHAPTER 6

CONCLUSION

This article presented a data driven approach to predict ETAs on freight rail networks in an online setting. The online ETA generation problem is posed as a series of independent origin-destination ETA prediction problems to avoid bias in the training data of a single general model due to time varying features, and is tractable for rail networks due to relative network complexity. Plus, it is shown to demonstrate specificity with respect to distinct feature weights (i.e., relative importance) between origin-destination models. ETA prediction models are constructed using combinations of features and linear and nonlinear SVR algorithms. Compared to naive prediction based on historical median runtimes, average improvements of 14% and maximum improvements of over 21% are achieved by the best performing SVR models

Based on these findings, our future research steps include the following. Due to the large variance caused by recrews, we are interested in developing a data-driven classifier to preemptively classify trips that are likely to be recrewed. Such a classifier would be needed to implement ETA prediction algorithms that rely on a dataset free of this large source of variability. For the trains that are not likely to be recrewed, further improvements in the ETA accuracy are possible with the construction of additional targeted traffic features. The current feature set includes sufficient information on the number of likely meets and passes that a train will incur, but only indirectly quantifies the delay that will be experienced as a result. A naive version of meet/pass planning could be implemented to generate additional features for the current models. Data describing yard operations is not present in the datasets at hand, but the addition of this information could assist predictions made close to the yards, which were shown to exhibit decreased performance.

REFERENCES

[1] P. Murali, M. Dessouky, F. Ordóñez, and K. Palmer, "A delay estimation technique for single and double-track railroads," *Transportation Research Part E: Logistics and Transportation Review*, vol. 46, no. 4, pp. 483–495, 2010.

[2] O. Wyman, "Assessment of european railways: Characteristics and crew-related safety," June 2016.

[3] M. J. Vromans, R. Dekker, and L. G. Kroon, "Reliability and heterogeneity of railway services," *European Journal of Operational Research*, vol. 172, no. 2, pp. 647–665, 2006.

[4] P. Kecman and R. M. Goverde, "An online railway traffic prediction model," in *RailCopenhagen2013: 5th International Conference on Railway Operations Modelling and Analysis, Copenhagen, Denmark, 13-15 May 2013*. International Association of Railway Operations Research (IAROR), 2013.

[5] C. Systematics, *National rail freight infrastructure capacity and investment study*. Cambridge Systematics, 2007.

[6] B. A. Weatherford, H. H. Willis, D. S. Ortiz, L. T. Mariano, J. E. Froemel, and S. A. Daly, *The State of US Railroads: A Review of Capacity and Performance Data*. Rand Corporation, 2008.

[7] Association of American Railroads, "Class I railroad statistics," February 2013.

[8] M. Dingler, Y.-C. Lai, and C. Barkan, "Impact of train type heterogeneity on single-track railway capacity," *Transportation Research Record: Journal of the Transportation Research Board*, no. 2117, pp. 41–49, 2009.

[9] M. Dingler, A. Koenig, S. Sogin, and C. P. Barkan, "Determining the causes of train delay," in *AREMA Annual Conference Proceedings*, 2010.

[10] A. Higgins, E. Kozan, and L. Ferreira, "Modelling delay risks associated with train schedules," *Transportation Planning and Technology*, vol. 19, no. 2, pp. 89–108, 1995.

[11] Q. Lu, M. Dessouky, and R. C. Leachman, "Modeling train movements through complex rail networks," *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, vol. 14, no. 1, pp. 48–75, 2004.

[12] S. Mu and M. Dessouky, "Scheduling freight trains traveling on complex networks," *Transportation Research Part B: Methodological*, vol. 45, no. 7, pp. 1103–1123, 2011.

[13] J. H. Hertenstein and R. S. Kaplan, *Burlington Northern: the ARES decision (a)*. Harvard Business School, 1991.

[14] S. F. Hallowell and P. T. Harker, "Predicting on-time performance in scheduled railroad operations: methodology and application to train scheduling," *Transportation Research Part A: Policy and Practice*, vol. 32, no. 4, pp. 279–295, 1998.

[15] D. R. Kraay and P. T. Harker, "Real-time scheduling of freight railroads," *Transportation Research Part B: Methodological*, vol. 29, no. 3, pp. 213–229, 1995.

[16] R. M. Goverde, *Punctuality of railway operations and timetable stability analysis*. TU Delft, Delft University of Technology, 2005.

[17] R. M. Goverde and L. Meng, "Advanced monitoring and management information of railway operations," *Journal of Rail Transport Planning & Management*, vol. 1, no. 2, pp. 69–79, 2011.

[18] A. D'Ariano and M. Pranzo, "An advanced real-time train dispatching system for minimizing the propagation of delays in a dispatching area under severe disturbances," *Networks and Spatial Economics*, vol. 9, no. 1, pp. 63–84, 2009.

[19] A. H. Lovett, C. T. Dick, and C. P. Barkan, "Determining freight train delay costs on railroad lines in North America," *Proceedings of RailTokyo 2015*, 2015.

[20] R. Estes and L. Rilett, "Advanced prediction of train arrival and crossing times at highway-railroad grade crossings," *Transportation Research Record: Journal of the Transportation Research Board*, no. 1708, pp. 68–76, 2000.

[21] E. Petersen and A. Taylor, "A structured model for rail line simulation and optimization," *Transportation Science*, vol. 16, no. 2, pp. 192–206, 1982.

[22] İ. Şahin, "Railway traffic control and train scheduling based oninter-train conflict management," *Transportation Research Part B: Methodological*, vol. 33, no. 7, pp. 511–534, 1999.

[23] M. Marinov and J. Viegas, "A mesoscopic simulation modelling methodology for analyzing and evaluating freight train operations in a rail network," *Simulation Modelling Practice and Theory*, vol. 19, no. 1, pp. 516–539, 2011.

[24] A. A. Assad, "Models for rail transportation," *Transportation Research Part A: General*, vol. 14, no. 3, pp. 205–220, 1980.

[25] K. B. Bonsra and J. Harbolovic, "Estimation of run times in a freight rail transportation network," M.S. thesis, Massachusetts Institute of Technology, 2012.

[26] R. Wang and D. B. Work, "Data driven approaches for passenger train delay estimation," in *Intelligent Transportation Systems (ITSC), 2015 IEEE 18th International Conference on*. IEEE, 2015, pp. 535–540.

[27] H. Li, D. Parikh, Q. He, B. Qian, Z. Li, D. Fang, and A. Hampapur, "Improving rail network velocity: A machine learning approach to predictive maintenance," *Transportation Research Part C: Emerging Technologies*, vol. 45, pp. 17–26, 2014.

[28] M. F. Gorman, "Statistical estimation of railroad congestion delay," *Transportation Research Part E: Logistics and Transportation Review*, vol. 45, no. 3, pp. 446–456, 2009.

[29] A. D'Ariano, M. Pranzo, and I. A. Hansen, "Conflict resolution and train speed coordination for solving real-time timetable perturbations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 2, pp. 208–222, 2007.

[30] H. Khadilkar, S. Salsingikar, and S. K. Sinha, "A machine learning approach for scheduling railway networks," in *Proceedings of the 7th International Conference on Railway Operations Modelling and Analysis, RailLille2017*. International Association of Railway Operations Research (IAROR), April 2017.

[31] B. Chen and P. T. Harker, "Two moments estimation of the delay on single-track rail lines with scheduled traffic," *Transportation Science*, vol. 24, no. 4, pp. 261–275, 1990.

[32] X. Chapuis, "Arrival time prediction using neural networks," in *Proceedings of the 7th International Conference on Railway Operations Modelling and Analysis, RailLille2017*. International Association of Railway Operations Research (IAROR), April 2017.

[33] F. M. B. A. Furtado, F. M. Bastos et al., "US and European freight railways: The differences that matter," in *Journal of the Transportation Research Forum*, vol. 52, no. 2. Transportation Research Forum, 2013, pp. 65–84.

[34] H. Pouryousef, P. Lautala, and T. White, "Railroad capacity tools and methodologies in the US and Europe," *Journal of Modern Transportation*, vol. 23, no. 1, pp. 30–42, 2015.

[35] Amtrak, "Route on-time performance," May 2016. [Online]. Available: https://www.amtrak.com/historical-on-time-performance

[36] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2001.

[37] A. Smola and S. Vishwanathan, *Introduction to Machine Learning*. Cambridge, United Kingdom: Cambridge University Press, 2008.

[38] H. Drucker, C. J. Burges, L. Kaufman, A. Smola, V. Vapnik et al., "Support vector regression machines," *Advances in neural information processing systems*, vol. 9, pp. 155–161, 1997.

[39] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[40] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[41] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, no. 2, pp. 301–320, 2005.

[42] C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[43] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*. ACM, 1992, pp. 144–152.

[44] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.