Estimating Train Delays in a Large Rail Network Using a Zero Shot Markov Model

Ramashish Gaurav
Nutanix Technologies India Pvt Ltd
Email: ramashish.gaurav@nutanix.com

Biplav Srivastava
IBM T J Watson Research Center, USA
Email: biplavs@us.ibm.com

***Abstract*—India runs the fourth largest railway transport network size carrying over 8 billion passengers per year. However, the travel experience of passengers is frequently marked by delays, i.e., late arrival of trains at stations, causing inconvenience. In a first, we study the systemic delays in train arrivals using n-order Markov frameworks and experiment with two regression-based models. Using train running-status data collected for two years, we report on an efficient algorithm for estimating delays at railway stations with near accurate results. This work can help railways to manage their resources, while also helping passengers and businesses served by them to efficiently plan their activities.**

I. Introduction

Trains have been a prominent mode of long-distance travel for decades, especially in the countries with a significant land area and large population. India, with a population of 1.324 billion people in 2016, has a railway system of network route length of $66,687$ kilometers, with $11,122$ locomotives, $7,216$ stations, that served 8.107 billion ridership in 2016[1].The Indian railway system is fourth largest in the world in terms of network size. However its trains are plagued with endemic delays that can be credited to (a) obsolete technology, e.g., dated rail engines, (b) size, e.g., large network structure and high railway traffic, (c) weather, e.g., fog in winter months in north India and rains during summer monsoons countrywide.

In this paper, we take the initial steps in understanding and predicting train delays. Specifically, we focus on the delays of trains, totaling 135, which pass through the busy Mughalsarai station (Station Code: MGS), over a two year period. We build an N-Order Markov Late Minutes Prediction Framework (N-OMLMPF) which, as we show, predicts near accurate late minutes at the stations the trains travel to. To the best of our knowledge, this is the first effort to predict train delays for Indian rail network. The closest prior work is by Ghosh et al. [3] [4] who study the structure and evolution of Indian Railway network, however, they do not estimate delays. Our analysis is complementary and agrees with the characteristics of the busiest train stations that they find. Masoud et al. [7] conducted a similar research for Iranian Railways, where they employ Neural Network models to learn and predict month-wise averaged late minutes (discretized into intervals) for trains from year 2005 to 2009. Our work is significantly different from theirs where we predict continuous late minutes along each of the stations during a journey. We now define the problem, outline contributions, and present our approach.

Problem Statement: Given a train and its route information, predict the delay in minutes at an in-line station during its journey on a valid date.

A. Contributions

Our main contributions are that we (also see [2]):

- as a first, present the dataset of 135 Indian trains' running status information (which captures delays along stations), collected for two years. The data is made public [2].
- build a scalable, train-agnostic, and Zero-Shot competent framework for predicting train arrival delays, learning from a fixed set of trains and transferring the knowledge to an unknown set of trains.
- study delays using n-order Markov Process Regression models and do Akaike Information Criterion (AIC) and Schwartz Bayesian Information Criterion (BIC) analysis to find the correct order of the Markov Process. We observed experimentally that most of the 135 trains followed a 1-order Markovian Process.
- discuss how the train-agnostic framework can leverage different types of trained models and be deployed in real time to predict the late minutes at an in-line station.

The rest of paper is arranged as follows. We first discuss the data about train operation and its analysis in Section II and then present the proposed model in Section III. Next, in section IV, we outline the experiments conducted with two different regression models: Random Forest Regression and Ridge Regression and give an exhaustive analysis of our results. Finally, we conclude with pointers for future research.

II. Data Preprocessing and Analysis

This section gives details of train information we collected for a span of two years[3]. Table I gives the statistics.

A. Data Collection and Segregation

We considered 135 trains that pass through Mughalsarai Station (MGS), one of top busiest stations in India. For them, we collected train running status information (*Train Data*) over

[1] http://www.indianrailways.gov.in/railwayboard/uploads/directorate/stat_econ/2015-2016/Year%20Book%20English%202015-16.pdf

[2] https://github.com/R-Gaurav/train-delay-estimation-ITSC2018/blob/master/Train_Delay_Estimation_Data_March_2016_February_2018.tar

[3] https://railwayapi.com

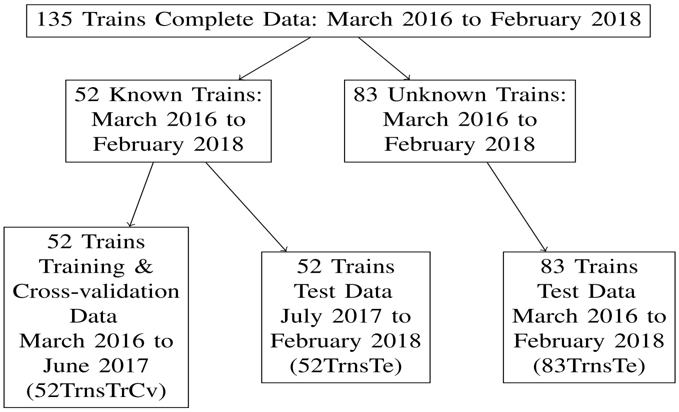

Fig. 1. Segregation of Complete Data of 135 Trains for Experimentation. The complete data is divided into two sets: 52 *Known Trains* and 83 *Unknown Trains*. *Known Trains* data is further subdivided into 52 Trains Training & Cross-validation Data (52TrnsTrCv) and 52 Trains Test Data (52TrnsTe) with different time periods. The *Unknown Trains* data (83TrnsTe) is kept intact to assess knowledge transfer from *Known Trains* to *Unknown Trains*.

the period of March 2016 to February 2018. A train's *Train Data* consists of multiple instances of journeys, where each journey has the same set of in-line stations that the train plies through. Table II has important fields of interest in *Train Data*.

Due to the infrequent running of trains, the amount of data collected for each of the trains greatly varied. Using the file size as criterion, we selected *Train Data* of 52 frequent trains (henceforth mentioned as *Known Trains*), out of 135, as training data. The data of remaining 83 trains (henceforth mentioned as *Unknown Trains*) were used for testing and evaluating the transfer of knowledge through trained models. Figure 1 pictorially illustrates the actual segregation of collected *Train Data* from March 2016 to February 2018 for 135 trains. One may recall that in traditional machine learning, the training and test data are drawn from the same set (or class). In contrast, we train our models on a seen set of *Known Trains* and test it on an unseen set of *Unknown Trains*, thus employing zero data of *Unknown Trains* for training, hence the term Zero-Shot. This problem setting is similar to Zero Shot Learning [6] where training and test set classes' data are disjoint. Fig.2 shows a train journey and related notations used in this paper.

TABLE I
DATA STATISTICS FOR 135 TRAINS COMPLETE DATA

Total number of trains considered	135
Total number of unique stations covered	819
Maximum number of journeys made by a train	334
Average number of journeys made by a train	48
Maximum number of stations in a train's route	129
Average number of stations in a train's route	30

TABLE II
DESCRIPTION OF *Train Data* COLLECTED FOR EACH TRAIN

Field Name	**Description**
$actarr\_date$	Actual arrival date of train at a station e.g. 19 Sep 2016
$station\_code$	Station code name (acronym) for a station e.g. MGS
$latemin$	Late minutes (arrival delay) at station e.g. 107
$distance$	Distance of a station from the source in kilometers e.g. 204
$month$	Jan, Feb, Mar... extracted from $actarr\_date$
$weekday$	Mon, Tue, Wed... extracted from $actarr\_date$

B. Data Preparation

We define a *data-frame* as a collection of multiple rows with fixed number of columns. For our experiments we prepared two types of data-frames, with one type being a data-frame Table III for each station (henceforth mentioned as *Known Stations*, totaling 621 out of 819) falling in the journey route of *Known Trains* by extracting required information from *Train Data* Table II of respective trains (in whose route the station fell) to train the models. Another type consisted of only one data-frame Table IV capturing certain information of all 819 stations; irrespective of whether they are in-line to *Known Trains* or *Unknown Trains*. We divided the journey data in *52TrnsTrCv* Data in ratio 4 to 1 to train and cross-validate the models and prepared data-frame (Table III) for the chosen 80% journey data. However, we did not prepare any data-frames (Table III) for rest 20% of *52TrnsTrCv* Data, *52TrnsTe* Data and *83TrnsTe* Data, thereby leaving them in their native format of *Train Data* Table II.

C. Data Analysis

Here we analyze the most important factors which drive our learning and prediction algorithm. As observed in Fig. 3, the spikes in each month signify that the mean late minutes at a station varies monthly (the colored dots are the individual late minutes during the month). This premise was verified with similar graphs obtained for other trains and their in-line stations. In Fig.4, the dots represent the mean of late minutes at each in-line station during a train's journey in a particular month. In Fig.4 we can see that the mean late minutes increase during journey up-till station BBS and later it decreases. We observed similar graphs for other trains and found that partial sequences of consecutive in-line stations characterize the delays during a train's journey.

III. PROPOSED MODEL

In this section, we explain our proposed regression-based N-OMLMPF algorithm and its components. Regression is the task of analyzing the effects of independent variables (in a multi-variate data) on a dependent continuous variable and predicting it. In our setting, the independent variables are the ones mentioned in Table III and the dependent continuous variable to be predicted is the target late minutes (**$\mathbf{Stn_0}$_late_minutes**). Our regression experiments with low RMSE and significant accuracy under 95% Confidence Interval back our hypothesis to cast it as a Regression based problem. We used Random Forest Regressors (RFRs) and Ridge Regressors (RRs) as two types of individual regression models in N-OMLMPF to learn, predict, evaluate, and compare results.

For real-time deployment and scalability, we avoided building train-specific models. Hence we looked for entities which would help us to frame a train-agnostic algorithm as well as enable knowledge transfer from *Known Trains* to *Unknown Trains*. A train's route is composed and characterized by the *Stations* in-line in its journey. Significant delays along a route which has more number of busy stations can be expected compared to the ones having lesser number of busy stations.

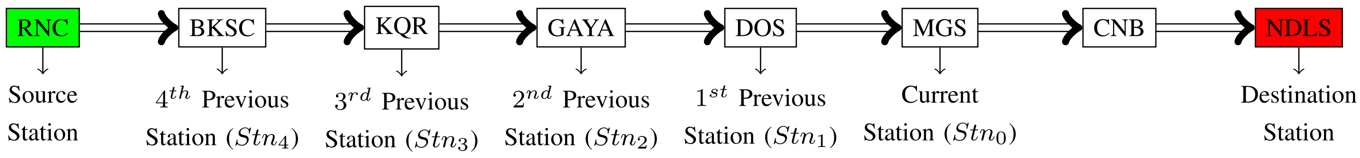

Fig. 2. Train Route of Train 12439: It starts at station RNC and ends at $NDLS$. For current station MGS, train arrival time at 4 previous stations are considered to prepare 4-$prev$-stn data-frame (Table III). Stn_i notation for i^{th} previous station is used throughout this paper.

TABLE III
DESCRIPTION OF TRAINING DATA PREPARED FROM *Train Data* TABLE II

train_type	**zone**	**is_superfast**	**month**	**weekday**
Is it Special, Express or Other?	What zone does the train belong to?	Is it super fast?	Month in which the journey is made	Weekday on which the journey is made
Obtained through train number[4] (e.g. 13050 for Train 13050)			Obtained from *actarr_date* (Table II)	

Stn$_1$_code	**...**	**Stn$_n$_code**	**late_mins_Stn$_1$**	**...**	**late_mins_Stn$_n$**	**db_Stn$_0$_Stn$_1$**	**...**	**db_Stn$_{n-1}$_Stn$_n$**
Station Code of Stn_1	...	Station Code of Stn_n	Late Minutes at Stn_1	...	Late Minutes at Stn_n	Distance between Stn_0 and Stn_1	...	Distance between Stn_{n-1} and Stn_n
Obtained from *station_code* (Table II)			Obtained from *latemin* (Table II)			Obtained from *distance* (Table II)		

Stn$_1$_dfs	**...**	**Stn$_n$_dfs**	**tfc_of_Stn$_1$**	**...**	**tfc_of_Stn$_n$**	**deg_of_Stn$_1$**	**...**	**deg_of_Stn$_n$**
Stn_1 distance from source station	...	Stn_n distance from source station	Traffic Strength of Stn_1	...	Traffic Strength of Stn_n	Degree Strength of Stn_1	...	Degree Strength of Stn_n
Obtained from *distance* (Table II)			Obtained from Open Government Data (OGD) [4]					

Stn$_0$_dfs	**Stn$_0$_tfc**	**Stn$_0$_deg**	**Stn$_0$_late_minutes**
Stn_0 distance from source station	Stn_0 traffic strength	Stn_0 degree strength	Current Station's target late minutes to be predicted
Obtained from *distance* (Table II)	Obtained from OGD [4]		Obtained from *latemin* (Table II)

Bold font indicates columns of prepared data-frame for a *Known Station*. We consider **Stn$_0$_late_minutes** to be a dependent feature. **tfc_of_Stn$_i$** and **deg_of_Stn$_i$** are the total number of trains passing through Stn_i and total number of direct connections of Stn_i to other stations, respectively. The data-frame is called n-$prev$-stn data-frame of a target station (Stn_0) for which it is prepared, where n is the number of previous stations.

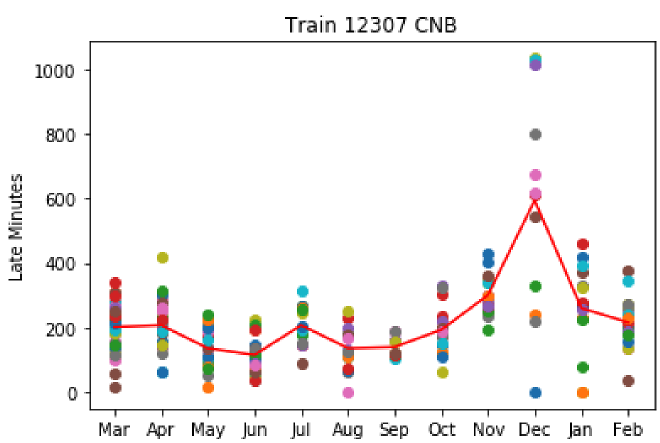

Fig. 3. Monthly variation of late minutes at station CNB for Train 12307

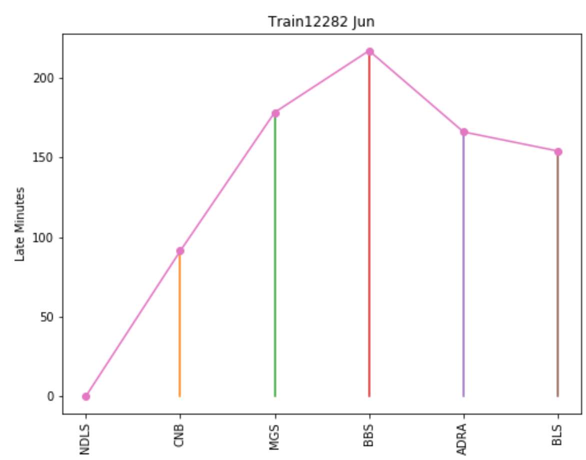

Fig. 4. Mean late minutes during Train 12282's journey in June 2017

Through the analysis of multiple figures similar to the ones mentioned in subsection II-C we observed the following *details* about the delay at in-line stations during a journey:

- It highly depends on the months during which the journey is made. One can observe the variations during summer (Jun in Fig.3) and winter months (Dec in Fig.3).
- Partial routes of consecutive *Stations* can be identified during journey which either increase or decrease the delay at next stations ($CNB \rightarrow MGS \rightarrow BBS$ in Fig.4).
- *Stations* with a high traffic and degree strength tend to be the bottleneck in a journey, thus increasing the overall lateness (MGS-a busy station in Fig.4.

Above points suggest that multiple deciding factors (e.g. the month of travel, the sequence of stations during a journey etc.) determine the late minutes at a station considered. Since we sought to use *Stations* to frame a train-agnostic late minutes prediction algorithm and for knowledge transfer, we prepare a data-frame Table III for each of the *Known Stations* capturing the *details* mentioned. Later, we train n-Order Markov Process Regression models for each *Known Station*; described next.

A. n-Order Markov Process Regression (n-OMPR) Models

The Markov Process asserts that the outcome at a current state depends only on the outcome of the immediately previous state. However if the current state's outcome depends on n

TABLE IV
DESCRIPTION OF *Station Features*

station	**latitude**	**longitude**	**stn_tfc**	**stn_deg**
Stn	Latitude	Longitude	Traffic Strength of station	Degree Strength of station
Obtained from $station\_code$	Obtained from Google Maps APIs		Obtained from OGD[5].	

The bold font texts are the columns in our prepared data-frame for collectively all 819 stations of *Known Trains* and *Unknown Trains*. **station** is used as a *key* to obtain rest 4 features on which k-NN is run. This data-frame helps to determine the semantically nearest station to a given station.

previous states, we call it an n-Order Markov Process. Here we assert that the late minutes at a current target station depends on the details of its n-previous stations (henceforth mentioned as $n\text{-}prev\text{-}stns$). This notion is effectively captured in data-frame Table III where we capture general features of a train, day and month of a journey and the characteristics of the $n\text{-}prev\text{-}stns$ along with that of the current target station. The idea is to learn n-OMPR models (Random Forest Regressors and Ridge Regressors) for each of the *Known Stations* using Algorithm 1 and later use those trained models to frame a train-agnostic late minutes prediction algorithm (N-OMLMPF Algorithm 2). Regression models are trained on each of the *Known Stations*' corresponding $n\text{-}prev\text{-}stn$ data-frame Table III with the values of n depending on the number of stations previous to it, subject to its positions during the journeys of multiple trains. The design is clarified in Section III-C of the extended paper [2].

B. k-Nearest Neighbor (k-NN) Search

Unknown Stations (*USs*) are the ones which along with the *Known Stations* (*KSs*) build the journey route of *Unknown Trains*. Since we made *Unknown Trains*' data Zero Shot, data-frame Table III is not prepared for *USs*. Thus we do not have n-OMPR models for them, hence we look for a KS which is best similar to the current target US with respect to features stated in Table IV; whose model could be used to approximate the predicted late minutes at the US. We employ k-NN search algorithm (Algorithm 3) to fulfill this objective. A two-step k-NN search is applied since latitude and longitude data are semantically different from traffic and degree strength data.

IV. EXPERIMENTS AND RESULT ANALYSIS

The N-OMLMPF Algorithm 2 was executed on three sets of data, namely Cross-validation Data of *Known Trains*, Test Data of *Known Trains* and Test Data of *Unknown Trains* as mentioned in Fig.1 for different values of N (in N-OMLMPF). We enumerate four detailed experiments below, which were conducted with both RFR and RR models individually:

1) Exp. 1: We ignored **tfc_of_Stn$_i$**, **deg_of_Stn$_i$** and **Stn$_i$_dfs** columns from data-frame Table III since these features are implicitly captured in **Stn$_i$_code**. Experiment was conducted on dataset *52TrnsTrCv*.
2) Exp. 2: We ignored the **Stn$_i$_code** columns from data-frame Table III as **tfc_of_Stn$_i$**, **deg_of_Stn$_i$** and **Stn$_i$_dfs** numerically capture the property of station codes. This was done for *Unknown Trains* case; because we did not have partial consecutive in-line station path of KSs and USs (hence no **Stn$_i$_code**s) due to the test data being Zero-Shot. The experiment was conducted on *83TrnsTe* data after learning the prediction models from *52TrnsTrCv* data to assess the transfer of knowledge from *Known Trains* to *Unknown Trains*.
3) Exp. 3: We conducted Exp. 2 again on *52TrnsTrCv* data, where results similar to that obtained in Exp 1 for cross-validation data endorses our notion of vice-versa representation of stations, as done in Exp. 1 and Exp. 2.
4) Exp. 4: We conducted Exp. 2 on *52TrnsTe* data with prediction models learned from *52TrnsTrCv* data.

Algorithm 1: Training n-OMPR Models

Input: List Of *Known Stations* (KS): $< KS_1, ...KS_M >$
Output: n-OMPR Models for *Known Stations*
for $i = 1; i <= 5; i+ = 1$ **do**
  $ips_{list} \leftarrow$ Initialize empty list (stores stations having i-OMPR models)
end
for $\forall\ stn_k \in$ *Known Stations* **do**
  for $i = 1; i <= 5; i+ = 1$ **do**
    df $\leftarrow$ Get stn_k's $i\text{-}prev\text{-}stn$ data-frame (Table III)
    if *df is not empty* **then**
      $mdl_i^{stn_k} \leftarrow$ Train RFR & RR Models on *df*
      $ips_{list} \leftarrow ips_{list} + stn_k$ ▷ Include stn_k in list
      Save $mdl_i^{stn_k}$
    end
  end
end
for $i = 1; i <= 5; i+ = 1$ **do**
  Save ips_{list}
end

After conducting the experiments we analyzed the results to evaluate the performance of trained models and to determine the optimum value of N (in N-OMLMPF). For brevity, we do not present the detailed results for all 135 trains, but we do justice by presenting 4-OMLMPF output on test data of one such train in Table V (negative numbers in the table suggest that the train arrived early by those many minutes).

A. Performance Evaluation of Models

We begin by noting again that a train's *Train Data* consists of multiple instances of journeys, where each journey has the same set of stations that the train plies through. For each in-line station during a train's journey, we calculated monthly 68%, 95%, and 99% Confidence Intervals (CI) around the mean of late minutes in a month, considering the train's complete *Train Data* with outlier late minutes removed by Tukey's Rule [5]. For each train's cross-validation/test *Train Data*, the percentage of the number of times the predicted late minutes for an in-line station fell under each matching CI was calculated. Then we averaged out all the percentages (calculated for each train) in different experiments enumerated above. Table VI shows the corresponding figures. In Table VII we present the mean Root Mean Square Error (RMSE) values for few *Known Trains* and few *Unknown Trains* obtained from their Test Data, where RMSE for a journey was calculated between the predicted late minutes and the actual late minutes.

Algorithm 2: N-OMLMPF for *Known Trains* and *Unknown Trains* (here the value of N is set as 3 $\implies$ limit the models up to 3-OMPR models)

Input: Train number tr_{num}, in-line stations list (stn_{jrny}), journey route information (Table II), ips_{list}
Output: A list (lms_{stn}) of predicted late minutes at each station during the journey
$lms_{stn} \leftarrow$ Initialize late minutes list with entry $< 0 >$ (0 minutes late at source)
for $i = 1; i < length(stn_{jrny}); i+ = 1$ **do**
  $crnt_{stn}$ = stn_{jrny}.At(i) ▷ Station at i^{th} position
  if $crnt_{stn}$ *is at position i = 1* **then**
    $df_{stn} \leftarrow$ Prepare $crnt_{stn}$'s 1-*prev-stn* row data-frame (Table III) using Table II with **late_mins_Stn$_1$** set as lms_{stn}.At(0)
    if $crnt_{stn} \notin 1ps_{list}$ **then**
      $crnt_{stn} \leftarrow$ Get nearest *Known Station* in $1ps_{list}$ using Algorithm 3
    end
    lms_{stn}.At(i) $\leftarrow$ Predict late minutes at $crnt_{stn}$ for df_{stn} using $mdl_1^{crnt_{stn}}$ model
  else if $crnt_{stn}$ *is at position i = 2* **then**
    $df_{stn} \leftarrow$ Prepare $crnt_{stn}$'s 2-*prev-stn* row data-frame (Table III) using Table II with **late_mins_Stn$_1$** set as lms_{stn}.At(1) and **late_mins_Stn$_2$** set as lms_{stn}.At(0)
    if $crnt_{stn} \notin 2ps_{list}$ **then**
      $crnt_{stn} \leftarrow$ Get nearest *Known Station* in $2ps_{list}$ using Algorithm 3
    end
    lms_{stn}.At(i) $\leftarrow$ Predict late minutes at $crnt_{stn}$ for df_{stn} using $mdl_2^{crnt_{stn}}$ model
  else
    ▷ $crnt_{stn}$ is at position i $\geq$ 3 during the journey
    $df_{stn} \leftarrow$ Prepare $crnt_{stn}$'s 3-*prev-stn* row data-frame (Table III) using Table II with **late_mins_Stn$_1$** set as lms_{stn}.At(i-1), **late_mins_Stn$_2$** set as lms_{stn}.At(i-2) and **late_mins_Stn$_3$** set as lms_{stn}.At(i-3)
    if $crnt_{stn} \notin 3ps_{list}$ **then**
      $crnt_{stn} \leftarrow$ Get nearest *Known Station* in $3ps_{list}$ using Algorithm 3
    end
    lms_{stn}.At(i) $\leftarrow$ Predict late minutes at $crnt_{stn}$ for df_{stn} using $mdl_3^{crnt_{stn}}$ model
  end
end

Algorithm 3: k-NN search framework to get a *Known Station* best similar to any type of Station (k set to 10)

Input: A Station stn_S, Valid ips_{list} of *Known Stations*
Output: A nearest *Known Station* stn_{KS}
$stn_{KS}^{nll} \leftarrow$ Get k-NN *Known Stations* to stn_S among stations in ips_{list} on the basis of Latitude and Longitude
$stn_{KS}^{ndt} \leftarrow$ Get k-NN *Known Stations* to stn_S among stations in stn_{KS}^{nll} on the basis of Degree and Traffic
Return the first station among stn_{KS}^{ndt}

It is to be noted that reported results in Table VI and VII are inclusive of journeys where the train actually got late at the source station, but these details could not be captured by our models due to their scarce occurrences.

Preliminary analysis of CI and mean RMSE observations showed that RFR models outperformed RR models. However, for sake of completion, we present CI observations of RR models for some selected experiments in Table VI. The scattering of individual late minutes at a station during a month; as observed in Fig.3 and other similar figures suggests to consider CI95 (or higher) since the late minutes are not closely centered around mean but cover a wider distribution around it. Under RFR Models column in Table VI, the figures in CI95 columns for Exp 1 and Exp 3 suggest that at an average we were able to predict late minutes at in-line stations during cross-validation journey data of *Known Trains* for approximately 62% times within 95% CI (say *accuracy* is 62%). Figures in Exp 2 under both RFR and RR Models columns in Table VI for *Unknown Trains*' test data do not seem promising, but since these results are for Zero-Shot trains for which significant amount of data is not available, the observations are appreciable. One should also note here the low mean RMSE values for *Unknown Trains* in Table VII. The higher *accuracies* (around 56% and 66% for CI95 and CI99) for *Known Trains*' test data in Exp 4 column under RFR Models column compared to that under RR Models column signify a very important conclusion. Random Forest Regressors (which are an ensemble of multiple decision trees) very well model the deciding factors (in Table III) compared to Ridge Regressors, thus the results state that the prediction of late minutes is effectively a decision-based regression task.

B. Determination of Optimum value of N in N-OMLMPF

We executed Algorithm 2 with values of $N \in$ (1..5), but which one truly captures the Markov Process property of delays along a train's journey? To answer this we employ two common model selection criterion [1]; Akaike Information Criterion (AIC) and Schwartz Bayesian Information Criterion (BIC) to choose the statistically best regression model.

$$AIC = n \times \ln\Big(\frac{SSE}{n}\Big) + 2p \tag{1}$$

$$BIC = n \times \ln\Big(\frac{SSE}{n}\Big) + p \times \ln(n) \tag{2}$$

where n stands for the number of observations used to train a model, SSE is the Squared Sum of Errors (between predicted late minutes and the actual late minutes) and p is the number of parameters in the model (number of columns in formatted data-frame Table III). Lower the score, better the model. The count of the number of times a run of N-OMLMPF (for a particular value of N) yielded the least AIC and BIC scores among all five runs for each train in all four experiments is noted in Table VIII. In Table VIII we see that delays along journey; for 40.38% to 67.30% of *Known Trains* under related experiments follow a 1-Order Markov Process since 1-OMLMPF scores minimum AIC and BIC score among other frameworks. Similarly 71.08% to 81.93% of *Unknown Trains* follow a 1-Order Markov Process. Rest of the trains follow a higher order Markov Process with diminishing indications. However lower cumulative RMSE scores (summed over all trains) obtained for 3- and 4-OMLMPF under different experimental settings suggest to use them for real-time deployment.

V. Conclusion and Future Work

Our objective was to predict the late minutes at an in-line station given the route information of a train and a valid date.

TABLE V
PREDICTED LATE MINUTES FOR *Known Train* 22811 TEST DATA (OBTAINED FROM 4-OMLMPF WITH RFR MODELS)

Stations:	**BBS**	**CTC**	**JJKR**	**BHC**	**BLS**	**KGP**	**BQA**	**ADRA**	**GMO**	**KQR**	**GAYA**	**MGS**	**CNB**	**NDLS**
Actual Late Minutes:	0	2	8	-1	13	25	19	18	2	9	-21	-5	6	15
Predicted Late Minutes:	0	2.75	6.83	0.01	17.44	16.52	11.22	17.65	1.94	16.01	-8.77	-0.25	12.26	23.10

TABLE VI
CONFIDENCE INTERVAL (CI) OBSERVATIONS FOR DIFFERENT EXPERIMENTS

	Random Forest Regressor (RFR) Models												**Ridge Regressor (RR) Models**					
	Exp 1 (Avg %age)			**Exp 2 (Avg %age)**			**Exp 3 (Avg %age)**			**Exp 4 (Avg %age)**			**Exp 2 (Avg %age)**			**Exp 4 (Avg %age)**		
	CI68	**CI95**	**CI99**	**CI68**	**CI95**	**CI99**	**CI68**	**CI95**	**CI99**	**CI68**	**CI95**	**CI99**	**CI68**	**CI95**	**CI99**	**CI68**	**CI95**	**CI99**
1-OMLMPF	34.65	61.37	70.47	5.90	14.73	18.51	33.67	61.05	70.21	27.60	55.41	65.57	4.97	12.87	17.29	22.34	44.30	55.71
2-OMLMPF	35.28	61.36	70.85	5.72	14.17	18.41	33.72	61.03	70.65	27.51	56.32	66.87	5.34	12.65	16.80	22.81	43.67	56.59
3-OMLMPF	33.86	62.31	71.42	6.00	14.79	18.81	33.80	62.13	71.58	27.81	55.89	66.98	4.89	12.46	16.76	22.21	44.05	55.67
4-OMLMPF	34.39	62.53	71.74	5.66	14.96	18.97	33.67	61.57	71.49	27.82	55.80	66.82	4.66	12.35	16.35	21.85	43.89	55.83
5-OMLMPF	34.77	62.70	72.10	5.51	14.52	18.75	33.45	62.03	71.96	27.93	56.20	67.07	4.61	12.43	16.16	21.85	43.87	55.18

CI68, **CI95**, and **CI99** respectively stand for 68% CI, 95% CI, and 99% CI. **Avg %age** stands for Average Percentage.

TABLE VII
MEAN RMSE VALUES FOR FEW *Known Trains* AND *Unknown Trains* TEST DATA (OBTAINED FROM 4-OMLMPF WITH RFR MODELS)

	Known Trains											**Unknown Trains**								
Trains	12305	12361	12815	12307	13131	13151	22811	22409	18612	13119	15635	03210	04401	04821	12141	12295	22308	12439	18311
Number of Journeys	16	14	39	84	19	83	28	14	47	25	13	2	1	6	3	4	28	2	3
Mean RMSE	87.12	89.38	96.61	88.26	62.84	82.34	53.71	44.72	29.42	80.66	80.22	57.37	23.86	31.97	53.38	68.49	44.83	11.75	36.20

Trains row consists of unique Train Numbers. **Number of Journeys** row denotes the number of journeys undertaken by the corresponding train in its Test Data. **Mean RMSE** row presents the average of the RMSEs of all journeys. For example, Train 12305 covered 16 journeys with a mean RMSE of 87.12.

TABLE VIII
BIC AND AIC ANALYSIS OF N-OMLMPF WITH RFR MODELS

	Random Forest Regressor Models							
	BIC Analysis				**AIC Analysis**			
	Exp 1	**Exp 2**	**Exp 3**	**Exp 4**	**Exp 1**	**Exp 2**	**Exp 3**	**Exp 4**
1-OMLMPF	32	68	35	29	21	59	31	23
2-OMLMPF	7	7	9	14	9	12	9	10
3-OMLMPF	9	5	6	5	12	7	7	11
4-OMLMPF	4	3	1	4	8	2	3	6
5-OMLMPF	0	0	1	0	2	3	2	2

The figures in each cell denote the number of times an N-OMLMPF scored minimum score among other runs, e.g. in **BIC Analysis** column for **Exp 1**, **1-OMLMPF** scored minimum BIC score for 32 trains among other runs.

The significant *accuracy* results in Table VI for *Known Trains*' and *Unknown Trains*' data demonstrates the efficacy of our proposed algorithm for a highly dynamic problem. We also determine experimentally and statistically that the delays along journey for most of the trains follow a 1-Order Markovian Process, while other few trains follow a higher order Markovian Process. Reasonably low RMSE results obtained for *Unknown Trains* in Table VII show that we were able to transfer knowledge from *Known Trains* to *Unknown Trains*, also we were able to predict late minutes of future journeys after learning from past data. The N-OMLMPF algorithm is so designed that it can leverage different types of prediction models and predict delay at stations for any train, thus it is train-agnostic. With just 1.2% of total trains in India, our approach was able to cover more than 11.3% of stations, thereby illustrating scalability. There are many avenues for future work: (a) one can expand the data collection and extend the analysis to trains India-wide, (b) one can also explore other approaches like time series prediction and neural networks. In particular, Recurrent Neural Networks (RNN) have the property of memorizing past details and predicting the next state. The prediction of delays along stations is inherently dynamic which implicitly calls for an online learning algorithm to continuously learn the changing behavior of railway network and delays. Thus one can attempt to develop an Online RNN algorithm for it. One can also consider predicting delays of trains in other countries.

VI. ACKNOWLEDGMENT

We would like to thank Debarun Bhattacharjya for his help in statistically discovering the order of Markovian delays through mathematical equations. We also thank Nutanix Technologies India Pvt Ltd for computational resources.

REFERENCES

[1] D. J. Beal, "Information criteria methods in sas for multiple linear regression models," *SESUG Proceedings. Paper SA05.*, 2007.

[2] R. Gaurav and B. Srivastava, "Estimating Train Delays in a Large Rail Network Using a Zero Shot Markov Model," in *On Arxiv at: https://arxiv.org/pdf/1806.02825.pdf*, 2018.

[3] S. Ghosh, A. Banerjee, N. Sharma, S. Agarwal, N. Ganguly, S. Bhattacharya, and A. Mukherjee, "Statistical analysis of the indian railway network: a complex network approach," *Acta Physica Polonica B Proceedings Supplement*, vol. 4, no. 2, pp. 123–138, 2011.

[4] S. Ghosh, A. Banerjee, N. Sharma, S. Agarwal, A. Mukherjee, and N. Ganguly, "Structure and evolution of the indian railway network," in *Summer Solstice International Conference on Discrete Models of Complex Systems*, 2010.

[5] D. C. Hoaglin, B. Iglewicz, and J. W. Tukey, "Performance of some resistant rules for outlier labeling," *Journal of the American Statistical Association*, 1986.

[6] C. H. Lampert, H. Nickisch, and S. Harmeling, "Attribute-based classification for zero-shot visual object categorization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 3, pp. 453–465, Mar. 2014.

[7] Y. Masoud, K. M. M., and S. Masoud, "Railway passenger train delay prediction via neural network model," *Journal of Advanced Transportation*, vol. 47, no. 3, pp. 355–368.