Leveraging Spark and Docker for Scalable, Reproducible Analysis of Railroad Defects

Dylan Chapp
University of Delaware
dchapp@udel.edu

Surya Kasturi
University of Delaware
suryak@udel.edu

ABSTRACT

The resiliency of railroad networks depends on the ability of railroad engineers to identify and mitigate track and rail defects. As railroads modernize their defect identification measures, the volume and velocity of defect data substantially increases, necessitating adoption of techniques for "Big Data" analytics. We present a study of railroad defect prediction built atop Apache Spark and Docker to achieve scalability and reproducibility.

1. INTRODUCTION

According to the United States Federal Railroad Administration Office of Safety Analysis, track defects are the second leading cause of accidents on railways in the United States. In light of the economic significance of railway accidents [1], there is a pressing need in the railroad engineering community to adopt data-driven scalable data analysis tools from the greater "Big Data" ecosystem. [3] Track maintenance–i.e., identifying and repairing defects–is one of the primary factors that affect the service life of a rail track, but due to the severe safety implications of undetected or unprepaired defects, the ability to predict common defects is highly desirable.

In this work, we present a case study centered on the analysis of two railroad defect data sets obtained from railroad engineering researchers in the University of Delaware Department of Civil Engineering. Hereafter we will refer to these datasets as the `rail_defects` data set and the `track_geometry_defects` data set. Respectively, these data sets describe defects in the rails themselves, such as voids or internal changes in crystalline structure, and misalignment of track components, such as one rail tilting away from the other. [3] We investigate the feasibility of predicting the type of a defect based on associated data such as geographic region, mean gross tonnage (MGT) the track is subject to, and rail type. In the rest of this paper, we outline the construction of our analysis platform, present some initial results on classification accuracy, and propose extensions to our work.

© 2016 ACM. ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

2. METHODOLOGY

Both of the data sets we target have mixed categorical and numerical features and $> 99\%$ of the individual defect records have a class label indicating the type of defect. In the case of `rail_defects`, there are 20 distinct defect types. For `track_geometry_defects`, there are 25 defect types. In light of these properties, we focus on the multilabel classification task for each data set. We decompose the task into a pipeline of three parts: preprocessing, training, and testing. We implement this pipeline using the MapReduce framework Apacahe Spark [2] and its parallel machine learning library MLLib, and package the data and analysis scripts as a Docker container for ease of dissemination. In the remainder of this section, we describe the pipeline components, also display in Figure 1

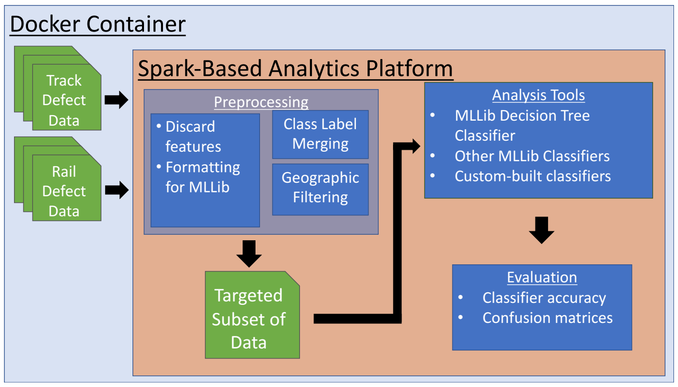

Figure 1: Block Diagram of Analytics Platform

We consider two stages of mandatory preprocessing. The first stage discards all columns except for a specified set, then discards any rows that are missing values for features from that set. The second stage maps the raw record strings to the format the MLLib API specifies, a key-value pair whose key is the type of defect and whose value is a feature vector.In addition to the above preprocessing, we implemented two optional stages: one to restrict the data to a geographically coherent region, and another to map each data point's class label to a "super- class" label indicating the general kind of defect (e.g., a welding-related defect, rather than one of the five kinds of specific welding defects). In our evaluation section, we demonstrate the usefulness of these additional preprocessing stages.

To build and evaluate our classifier, we split the subset of data remaining after preprocessing into training and testing

Table 1: Rail Defects Mapping

Super Class	Defect Types
T	TDD, TDC, TD, TDT
B	BRO, BHB, BB
W	HW, HWJ, OAW, SW
Others	SD, VSH, HSH, TW, CH, FH, PIPE, DR, EFBW

sets consisting of, respectively, 70% and 30% of the original data. Membership in the training and testing sets is determined by uniform random sampling. We then train an instance of MLLib's decision tree classifier on the training set and test its predictions. In principle, any other MLLib classifier with a compatible API could be trained instead, but we elected to keep our classifier type fixed and investigate the effect of the "class-merging" and "geographic filtering" preprocessing steps on accuracy.

3. EVALUATION

To evaluate our classifier's performance, we examine the overall accuracy rate of the classifier and its associated confusion matrix. When we trained the classifier on training data drawn uniformly at random from `rail_defects` dataset with each defect type as a class label, the classifier predicted with an accuracy of 42.55%.

3.1 Class Label Merging

We propose mapping each data point's class label to a "super-class" label indicating its general kind. Out of 20 defect types in `rail_defects` dataset, 11 are mapped to 3 three "super-classes". Table 3.1 shows mapped and unmapped defect types.

With this mapping, we show that the prediction accuracy of rail defects is improved using a hierarchical classification scheme . First a classifier is trained to decide super-class of data point, then a second classifier is used to predict its defect type. When this model applied on the training data, the classifier predicted with an accuracy of 89.90%. Figure 2 shows the confusion matrix of the respective result.

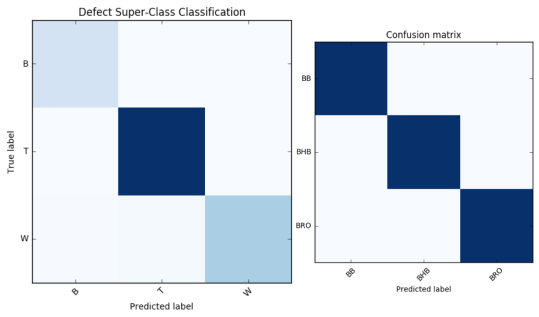

Figure 2: Confusion Matrix of the hierarchical classification scheme

3.2 Geographic Filtering

We propose that if subdivisions have similar numbers of each kind of defect, then we should group these subdivisions' data points and train a classifier with the expressed purpose of achieving good accuracy for that set of subdivisions. To determine which subdivisions to merge, we propose computing the χ-squared distance S defined below for each pair of subdivisions, then merge them based on a fixed threshold.

$$S(D_1, D_2) = \sum_{i=0}^{N} \frac{(x_i - y_i)^2}{(x_i + y_i)}$$

We demonstrate the potential of grouping based on defect type distributions below. We compute $S(x, y)$ for each pair of subdivisions within the Appalachian division and display the results in the heat map in Figure 3.2

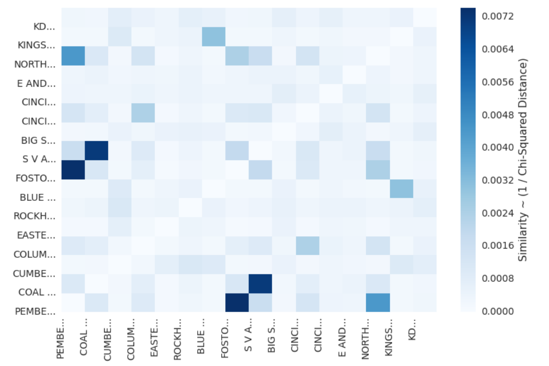

Figure 3: χ-squared distance between defect distributions for each subdivision

4. CONCLUSIONS AND CONTINUING WORK

We identified the potential of a hierarchical classification stage to improve the accuracy of defect type predictions. Additionally, we determined while that merely training classifiers on data from geographically-similar regions does not yield a significant improvement in accuracy, attempting to group together regions whose defect distributions are similar may prove useful.

Future directions for this work include evaluating classifiers beyond decision trees, and refining the similarity metric on defect distributions we use to group regions.

5. REFERENCES

[1] D. H. Schafer. A prediction model for broken rails and an analysis of their economic impact. *2008 AREMA Conference*, 2008.

[2] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, and I. Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, NSDI'12, pages 2–2, Berkeley, CA, USA, 2012. USENIX Association.

[3] A. M. Zarembski. Some examples of big data in railroad engineering. In *2014 IEEE International Conference on Big Data (Big Data)*, pages 96–102, Oct 2014.