REPORT CDOT-2022-09 December 2022

Data-Driven Bridge Management Using Descriptive and Predictive Machine Learning Models

APPLIED RESEARCH & INNOVATION BRANCH

Dr. Farnoush Banaei-Kashani
Dr. Kevin Rens

COLORADO
Department of Transportation

The contents of this report reflect the views of the author(s), who is(are) responsible for the facts and accuracy of the data presented herein. The contents do not necessarily reflect the official views of the Colorado Department of Transportation or the Federal Highway Administration. This report does not constitute a standard, specification, or regulation.

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
CDOT-2022-09		

4. Title and Subtitle	5. Report Date
Data-Driven Bridge Management Using Descriptive and Predictive Machine Learning Models	December 2022
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Farnoush Banaei-Kashani, Kevin Rens	

9. Performing Organization Name and Address	10. Work Unit No. (TRAIS)
University of Colorado Denver 1201 Larimer St, Denver, CO 80204	11. Contract or Grant No. XXX.XX

12. Sponsoring Agency Name and Address	13. Type of Report and Period Covered
Colorado Department of Transportation - Research 2829 W. Howard Pl. Denver CO, 80204	Final
	14. Sponsoring Agency Code

15. Supplementary Notes

Prepared in cooperation with the US Department of Transportation, Federal Highway Administration

16. Abstract

Bridges deteriorate with time and use. To monitor deterioration of the bridges, several US Acts mandate the state and local governmental agencies (including cities, state transportation agencies, etc.) to perform regular bridge inspections. The aforementioned inspections across the nation, which have been conducted since 1970's (including our region), have generated valuable historic databases of bridge data based in local and state governmental agencies. While these agencies currently use these inspections to prevent failure and to administrate the national bridge network by setting priorities and establishing criteria to allocate available resources to the structures in most critical conditions, we believe these databases are heavily underutilized. In particular, with the advent of machine learning and data mining methods, we envision data-driven solutions that can derive much more valued hidden knowledge that can be utilized for enhanced bridge management.

While in the past, various data-driven deterioration models are proposed in the literature to model bridge deterioration, these models either suffer from low accuracy or are too complex to be applicable. Recently deep learning is shown to significantly outperform other analytical modeling methodologies in a variety of application domains. In this study, we present new deep learning models for enhanced bridge management. In particular we focus on the two problems of bridge subtyping (descriptive analysis) and bridge deterioration forecasting (predictive analysis). Through empirical evaluation with real data, we demonstrate that our solutions for these problems significantly enhance the state-of-the-art in bridge management.

17. Keywords	18. Distribution Statement
Bridge Deterioration Forecast, Bridge Family Generation, Deep Learning Model, Data-Driven	This document is available on CDOT's website https://www.codot.gov/programs/research

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified		

Acknowledgements

We gratefully acknowledge the support and guidance of the CDOT Research Project Manager Study Manager, Thien Tran (CDOT Applied Research and Innovations Branch), and the Study Panel: Jessica Martinez, (CDOT Bridge - Co- Champion); Natasha Butler (CDOT Bridge - Co-Champion); Lynn Crosswell (CDOT Bridger - Co-Champion); Michael Collins, (CDOT Bridge - Sponsor). It has been a privilege to work with these collaborative partners on such an impactful and rewarding project. We also thank Steve Cohn and David Reeves for their guidance and support throughout the project. Finally, we are thankful for the contributions of the graduate students at University of Colorado Denver: Tobby Lie, Khang Nguyen, Masoumeh Abolfathi, Rumana Sultana, and Ryan Cheng. This work was supported by CDOT contract 220.04.

Executive Summary

Since 1970's, several U.S. Acts have mandated all local and state transportation agencies across the nation to perform regular inspections of the bridges (and culverts) in the regions under their jurisdiction. These inspections have generated valuable historical databases of bridge performance data, which have remained considerably underutilized to date. In this project, with the advent of machine learning and data mining methods, we envisioned data-driven solutions that could derive valuable hidden knowledge from these databases, the knowledge that could be effectively utilized for enhanced bridge management. Toward this end, with this study we have developed advanced data-driven artificial intelligence (AI) models (namely, deep learning models) that can leverage the existing historical bridge (and culvert) performance data, as well as weather data and traffic data, to enable (1) accurate bridge deterioration forecasting (i.e., predictive analytics), and (2) effective bridge family generation or bridge subtyping (i.e., descriptive analytics). With extensive experimental evaluation using multi-modal real datasets including bridge performance data, traffic data and weather data for all bridges in Colorado, we have demonstrated that a selection of our proposed models significantly outperform existing models for the aforementioned two problems, respectively.

Bridge engineers in governmental transportation agencies need to regularly forecast deterioration condition of the bridges (and bridge families) under their supervision in order to develop bridge maintenance plans, and even more importantly, identify anomalous bridge deterioration that can result in bridge accidents. Accordingly, we have turned the deep learning models developed under this project into a software tool that can facilitate effective bridge management for Colorado Department of Transportation (CDOT) bridge engineers. Unlike existing bridge management tools such as BrM (i.e., the AASHTO sponsored Bridge Management software) used by most of the bridge engineers across the nation, to the best of our knowledge our proposed tool is the first to make accurate deterioration forecasts/predictions based on historical data, in a similar way weather forecasts are generated. This tool is developed as a standalone, platform-independent and user-friendly software application.

Implementation Statement

We have developed and delivered a tool for bridge deterioration forecasting and bridge family generation that can make accurate deterioration forecasts/predictions based on historical data. We have demonstrated accuracy of our proposed advanced data-driven AI models (namely, deep learning models) that enable this tool via extensive and rigorous experimental evaluation using multi-modal real datasets including bridge performance data, traffic data and weather data for all bridges in Colorado, and shown that a selection of our proposed models significantly outperform existing models for bridge deterioration forecasting and bridge family generation.

Accordingly, we recommend adoption of the developed tool by bridge engineers in state and local transportation agencies to enable further accurate and enhanced bridge deterioration forecasting, which in turn can improve the ability of these agencies for more cost-effective and efficient bridge management and maintenance planning.

Table of Contents

List of Figures

List of Tables

1. Introduction

Bridges deteriorate with time and use. The deterioration process is affected by several factors, such as materials, structural design and behavior, daily traffic, freeze and thaw cycles, climate, pollution, temperature variation [24-26]. After a certain period of time has elapsed, the deterioration processes accelerate and in a relatively short time interval the components can lose the capacity to carry the loads they were designed to support.

To address this national issue, several US Acts [27] mandate the state and local governmental agencies (including cities, state transportation agencies, etc.) to perform regular bridge inspections. These Acts define the requirements, periodicity, and procedures for such inspections in the US. Inspections are required to assess the extension, implications, and current state of deterioration processes that may exist, and they need to be performed at regular time intervals, typically 2 years. A bridge report is generated after each inspection. All bridge reports collect and offer specific data about health of the inspected bridge, including sufficiency rating, condition rating, structure identification, year built, average daily traffic, and average daily truck traffic. For example, condition ratings (aka condition indexes) are quantitative descriptors of the state of structure parts that can be used in the assessment for the structures maintenance [26, 27]. By associating a deteriorated state to a number, instead of using qualitative description of the state, much more flexibility can be achieved in monitoring groups of similar structures [28-33]. The adoption of condition ratings in the evaluation of structures allows consistency and uniformity, making it possible to compare structural performance, establish priorities, and also prevent failures and accidents.

The aforementioned inspections across the nation, which have been conducted since 1970's (including our region), have generated valuable historic databases of bridge data based in local and state governmental agencies. While these agencies currently use these inspections to prevent failure and to administrate the national bridge network by setting priorities and establishing criteria to allocate available resources to the structures in most critical conditions, we believe these databases are heavily underutilized. In particular, with the advent of machine learning and data mining methods, we envision data-driven solutions that can derive much more valued hidden knowledge that can be utilized for enhanced bridge management.

While in the past, various data-driven deterioration models including Bayesian models, Probit model, and Markov chains are proposed in the literature to model bridge deterioration [24, 25, 34-38], these models either suffer from low accuracy or are too complex to be applicable. Moreover, they only address the problem of deterioration forecasting. Recently deep learning is shown to significantly outperform other analytical modeling methodologies in a variety of application domains, such as computational biology, Electronic Health Record (EHR) data analysis, activity detection, scene labeling, image captioning, and object detection [39-47]. In the past, we have introduced and deployed various deep learning based models, e.g., for sleep stages classification using brain signals [48, 49], mobility monitoring [50, 51], and activity classification [52]. In this study, we propose to develop deep learning models for enhanced bridge management. In particular we focus on the two problems of *bridge subtyping* (descriptive analysis) and *bridge deterioration forecasting* (predictive analysis). Effective solutions for these problems will significantly enhance the state-of-the-art in bridge management.

Below we further elaborate on our two focus research problems:

1. Bridge Subtyping Tool for Descriptive Analysis: With this tool one can perform descriptive analysis of the bridges and their performance by (1) objectively categorizing bridges based on their quality and deterioration performance given static and dynamic features associated with each bridge per bridge inspection reports, (2) analyzing and determining the hidden links between bridge performance and bridge structural properties, utilization statistics, and environmental characteristics in each category, and finally (3) identifying the distinct features as well as characteristic behavioral and performance trends for bridges in each category. In turn, this knowledge can be effectively used not only to make more informed choices in maintenance and repair planning for existing bridges, but also to make better design choices in building new bridges.

2. Bridge Deterioration Forecasting Tool for Predictive Analysis: With this tool one can perform predictive analysis of the bridges by accurate prediction of quantitative descriptors for the structure deterioration state (e.g., condition ratings). Accurate prediction of these descriptors are not only crucial in establishing maintenance priorities

and performing proactive bridge monitoring with optimized resource allocation, but also more importantly essential for failure prevention.

Our proposed automated tools allow for enhanced bridge management by improving depth, accuracy, and efficiency/speed in descriptive and predictive analysis of the historic bridge data reported by bridge inspectors. In turn, this can lead into more effective resource allocation for bridge monitoring, maintenance, and construction.

In the remaining of this report, first in Section 2 we will present the data sources used in this study along with the data preprocessing tools developed to prepare the raw data for data-driven bridge deterioration forecasting modeling and bridge subtyping (aka bridge family generation) modeling. Thereafter, in Sections 3 and 4, we will define the problem in further detail, review the literature, present our proposed methods/models, and discuss our experimental evaluation of the proposed models for bridge deterioration forecasting and bridge family generation, respectively. Finally, in Section 5 we will conclude and briefly discuss future directions for this research.

2. Data Preparation

The data used in this work comes in the form of 3 modes: (1) Bridge evaluation data, (2) Traffic data, and (3) Weather data. This section will detail the datasets used, features selected, as well as a categorization of different types of data. Figure 1 shows how these datasets are aligned using location data.

2.1 National Bridge Inventory

For the bridge evaluation and traffic data, we utilized publicly available data from the National Bridge Inventory (NBI) [20] database. This dataset contains national bridge inspection/evaluation data collected over the years 1992-current. We extracted the following features from this data which served as inputs into our model for each year's worth of evaluations, the selected features are specified in Table 1, as well as a description of the color coding used to describe the nature of their data.

The frequency of collection for the NBI dataset is on an annual basis. There are a total of 134 different features available in the NBI dataset. Surveying domain experts was leveraged to intelligently select and trim down the total features for model training. This work had access to domain experts on our team as well as in the Colorado Department of Transportation, and thus, essential feedback was gathered there to best select relevant features. An emphasis on the importance of traffic statistics for predicting bridge structure deterioration was expressed by domain experts justifying the need for ADT and Traffic Lanes on. The features selected also contain the condition ratings of interest to be predicted (Deck Condition Rating, Superstructure Condition Rating, and Substructure Condition Rating). The prepared NBI dataset is available here: https://github.com/tobby-lie/public-cdot-report-code/blob/main/final_outfile/NBI_Final.xlsx

Feature	Description	Data Type	Source
ADT	Number that shows the average daily traffic volume of a bridge structure.	Float	NBI
Traffic Lanes on	Number of traffic lanes on a bridge structure.	Integer	NBI
Deck Condition Rating	Number describing the overall condition rating of the bridge deck. Number from 1 to 9.	Integer	NBI
Superstructure Condition Rating	Number describing the physical condition of all structural members. Number from 1 to 9.	Integer	NBI
Substructure Condition Rating	Number describing the physical condition of piers, abutments, piles, fenders, and footings. Number from 1 to 9.	Integer	NBI
Culvert Rating	Number describing the overall condition rating of the culvert. Number from 1 to 9.	Integer	NBI

Table 1: Features Selected form NBI Dataset (Yellow Indicates Traffic Input Data, Blue Indicates Bridge Evaluation Input Data, And Green Indicates Bridge Evaluation Input/Output Data)

2.2 National Oceanic and Atmospheric Administration

For the weather data, we utilized another publicly available dataset from the National Oceanic and Atmospheric Administration's (NOAA) online weather database, the dataset is the Normals Daily dataset [21]. This dataset contains daily precipitation and snow records over global land areas. The following features from this data specified in Table 2 were used.

Feature	Description	Data Type	
Precipitation	Daily precipitation in millimeters.	Float	NOAA
Snowfall	Daily snowfall in millimeters.	Float	NOAA
Snow Depth	Daily snow depth in millimeters	Float	NOAA

Table 2: Features Selected from NOAA Dataset (Orange Indicates Weather Input Data)

Once again, domain experts were consulted to determine relevant features for the task of bridge deterioration forecasting. There was an emphasis on the importance of precipitation regarding predicting bridge deterioration forecasting. This is because protective epoxy coatings may be affected negatively when moisture is introduced to a bridge structure. In addition to this, there was also an importance in considering snowfall and snow depth since the freeze/thaw cycles of bridge structures contribute to how bridges may deteriorate as well as how maintenance strategies are scheduled. The prepared NOAA dataset is available here: https://github.com/tobby-lie/public-cdot-report-code/blob/main/NOAA_final_outfile/NOAA_NBI_Final.xlsx

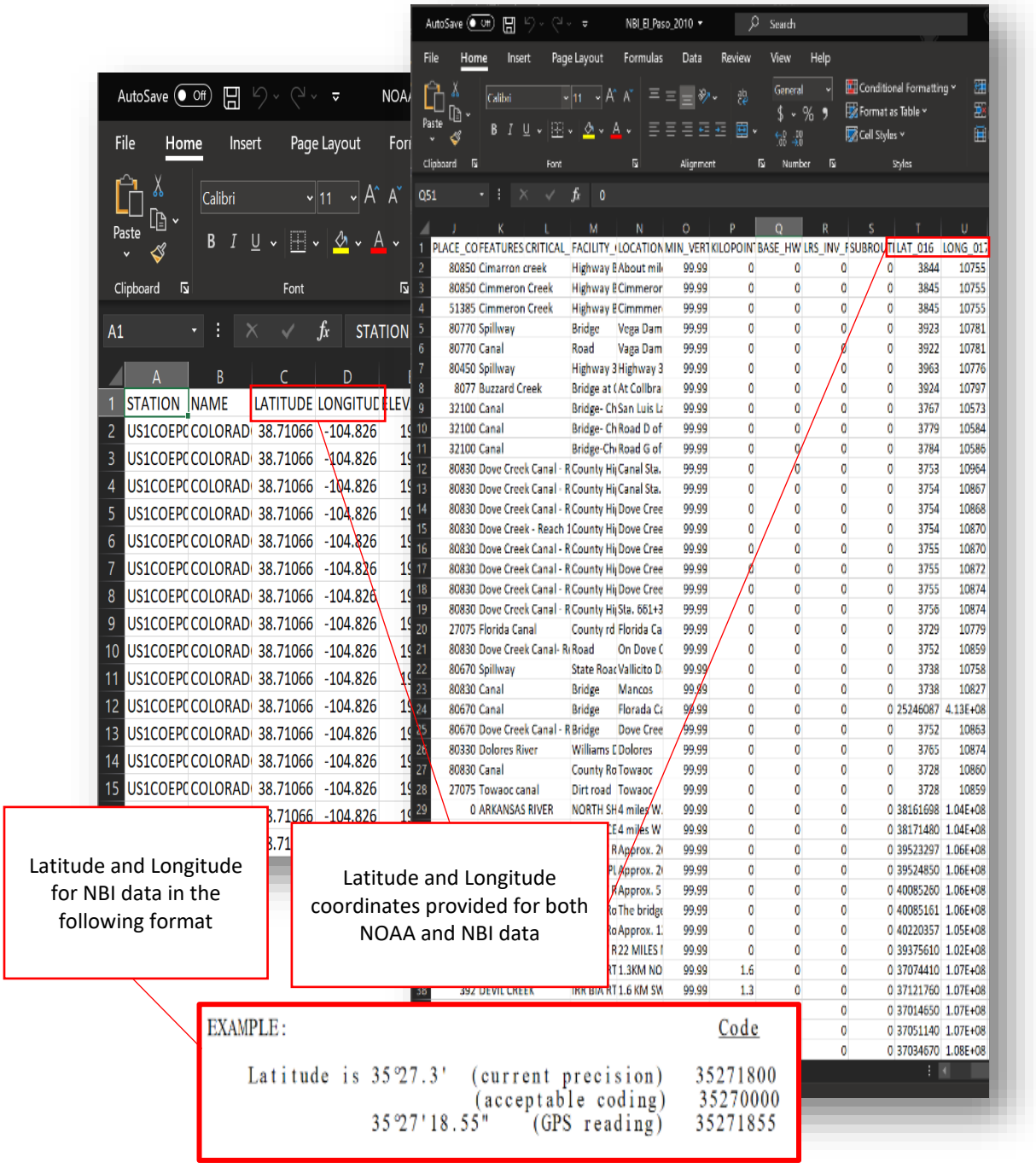

Figure 1: Illustration of NOAA and NBI Datasets Showing Coordinates used for Localization

3. Bridge Deterioration Forecasting

3.1 Problem Definition

The focus of this work revolves around effective multivariate time-series forecasting of bridge condition ratings over time with the utilization of physics guidance. Given a set of consecutive time steps *t*, each of which are multivariate with length *m*, we wish to predict *N* time steps forward into the future.

The input for this problem can be represented as $X = \begin{bmatrix} x_{11} & x_{12} & x_{1n} \\ \vdots & \vdots \cdots & \vdots \\ x_{m1} & x_{m2} & x_{mn} \end{bmatrix}$, where *n* is the total number of input time steps and *m* is the total number of variables for each time step. The variables that exist for each input (x_{ij}) consist of the condition ratings of interest as well as optional supplemental data. We wish to train a multivariate time-series forecasting model f_θ that can predict 1 multivariate time step forward into the future past *n* from the input, the prediction step being of length *m*. The variables for the output are the same as the variables for the input. In addition to this, we would like to also condition our model using some form of physical grounding which is embedded into the model in some way. A sliding window technique can be applied to the prediction process to output more than one prediction step as shown in Figure 2. Also shown in this figure, the variables of interest from the output vector can be selected.

The success of the methods is evaluated via root mean squared error (RMSE). RMSE is defined as $\sqrt{\frac{\sum_{i=1}^{N}(x_i - \hat{x}_i)^2}{N}}$ where x_i is a ground truth value, $\hat{x}_i$ is a predicted value, *N* is the number of data points. The evaluation process measures the error in the produced outputs during evaluation of the test split of the data after model training. Each sample in the test split of the data is used to obtain an RMSE score for each individual rating, an average of RMSE scores is taken over all test split samples to obtain an aggregate RMSE measure, an example of this is shown in Figure 3 with a test split size of 2215 data samples, 21 total time steps, and 3 variables.

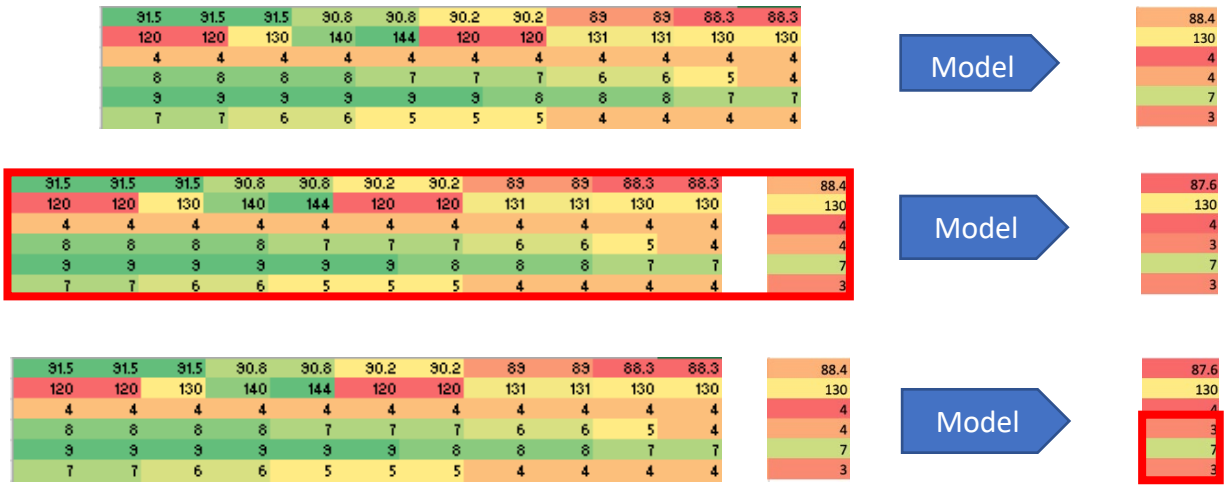

Figure 2: Illustration of Model Prediction Process

Figure 3: RMSE Measurement Process

To reiterate, we wish to train a multivariate time-series forecasting model f_θ such that it generates a single time step output with length m represents the number of variables in the output time step. This model is to be conditioned on input matrix X where its dimensions are n and m where n is the total number of input time steps and m is the number of variables in each time step. This model also needs to be conditioned using some form of physical grounding via the physics-guided neural network approaches. The success of this model will be evaluated using RMSE as the metric.

3.2 Literature Review

In this section, we explore some relevant work pertaining to purely data-driven models for bridge asset management. The work concerning this subsection pertains primarily to the

prediction or forecasting of future ratings indicating the structural condition of bridge structures. Various models are touched upon in the work below and some of the work inspires some of the methods and approaches presented in this paper.

Srikanth et al. provide a survey of different data-driven bridge deterioration models for deck condition rating forecasting, highlighting the advantages and limitations for each [3]. Deterministic models assume that the tendency of bridge deterioration processes is certain, and therefore, regression analysis is applicable for deterioration forecast. On the one hand, these models are simple and practical, and on the other hand they have limited accuracy as they neglect uncertainty due to inherent stochastic nature of infrastructure deterioration, while also computationally expensive to update. In contrast, stochastic models consider the bridge deterioration process as one or more random variables. These models are able to capture uncertainty, are compatible with existing qualitative/discrete bridge condition rating systems, and are simple and practical. However, such models cannot be used to assess the reliability of a structure in terms of strengths and stresses. Finally, mechanistic models have the capability to relate a qualitative measurement of condition state to the quantitative physical parameters of the bridge such as material properties, stress condition, and structural behavior. These models are suitable for project level analysis and they provide reliability based quantitative deterioration prediction for bridge elements. On the down side, mechanistic models require considerable amount of data which can be costly, and they cannot be directly integrated into a bridge management system due to the high cost of data collection. Artificial intelligence-based models utilize neural networks to predict the deterioration of concrete decks based on inspection records (most relevant to our work). An advantage of these models is that they can generate missing condition state data to fill in gaps due to irregular inspections. A limitation to these models is that neural networks are just an approach to artificially generate missing data and requires complementary tools to utilize the generated information for modeling bridge deterioration.

Liu et al. present a deep learning-based bridge condition data modeling approach which enables the prediction of future condition ratings of highway bridge components [4]. The work in this paper utilizes a deep Convolutional Neural Network (CNN) model trained on historical bridge inspection data such as the record items in the National Bridge Inventory dataset (NBI). The bridge components monitored refer to three primary bridge components: deck, superstructure, and substructure. They performed a case study using bridges from Maryland and

Delaware, taking 24 features from the NBI dataset for the years of 1992-2017. Input into their CNN model consisted of a matrix where each row was an individual year's worth of data, and each column represented a feature for each timestep. The outputs of their model were condition ratings for the immediate next timestep for bridge deck, superstructure, and substructure. After experimenting with the number of features and feature combinations in an automated way, they found that their model performed best when all 24 were utilized.

Morcous compares artificial neural networks against case-based reasoning for the task of predicting future condition of bridge/bridge components [5]. The objective of this comparison was to determine the pros and cons of the two approaches to guide transportation agencies in selecting which approach best suited their needs. Artificial neural network-based models were used in the following ways: (1) In a small-scale investigation where an artificial neural network was used to predict future conditions of bridge superstructures as a function of their age. (2) To predict sufficiency rating given several explanatory variables such as design type, material type, traffic volume, and age. (3) To predict the short-term future condition of pavement cracks given past condition ratings. Case-based reasoning models were used primarily to predict the future condition of bridge decks by reusing the recorded condition of other decks similar in physical features, environmental and operational conditions, and maintenance history. This work noted that the artificial neural network models are statistical and rely on generalized knowledge from a data distribution. This approach relies heavily on the quality of training data. On the other hand, case-based reasoning models are based on specialized knowledge, meaning that only input data that shares a high similarity to the data used to develop the model in the first place are viable for producing outputs. The author concludes that although case-based reasoning is very accurate, it is difficult to tune and set up and is not useful when examining new cases with data dissimilar to cases in its case library. On the other hand, the artificial neural network approach seemed less laborious but expanding and/or updating such a model requires more effort in its initial development. In addition, artificial neural networks require data to be transformed in some cases from symbolic to either binary or continuous, thus leading to loss of information.

Tokdemir et al. study artificial neural networks and genetic algorithms for predicting bridge sufficiency ratings [6]. Based on explanatory variables such as geometrical attributes, structure age, traffic volume, and structural attributes, artificial neural networks and genetic algorithms were used to predict sufficiency ratings. Ultimately, this work found that artificial

neural networks outperformed genetic algorithms when different models were constructed for different levels of sufficiency ratings and genetic algorithms outperformed artificial neural networks when using the entirety of the data. The main disadvantage of genetic algorithms, despite their success based on leveraging entire datasets, is that they have very long training times. They note that to alleviate the issue of long training times for genetic algorithms, it may be of some use to work with smaller datasets or benchmark genetic algorithms against only the best artificial neural network models in the interest of time if some comparison between models is necessary.

3.3 Proposed Methods

In this section, we will cover the modeling methods used or considered in the bridge deterioration forecasting solution. This will include the different models and categories of models that we explored for the problem of multi-variate time-series prediction. There are three categories of models that we explored: (1) Traditional regression-based models, (2) Baseline neural network models, and (3) Novel neural network models. The logic behind exploring these different categories was to find a true baseline (traditional regression-based), find a more complex class of models (baseline neural networks), and examine a class of models that would be more cutting edge even with respect to the more complex models (novel neural networks). In this section, the traditional regression-based models were found to be much less versatile, unable to obtain an aggregate understanding of many data samples while the neural networks added a layer of complexity which allowed for more versatility.

3.3.1 Traditional Regression Models

In this subsection, we will cover the first and most baseline class of models utilized in this work. This class refers to traditional regression models which are simpler than the neural network architectures touched on in further chapters.

3.3.1.1 Vector Autoregression

Vector Autoregression (VAR) is a modeling method used for multivariate forecasting [7]. It is used when you have at least two time series or variables that have an influence on one another. Autoregressive refers to each variable in each time series being modeled based on its past values. In autoregressive models, each variable is modeled using a linear combination of its

lags (past observations). In VAR specifically, each variable is modeled using a linear combination of its own lags as well as lags of other variables. An autoregression model for a single variable looks like the following, where α is the constant intercept, β_i are the coefficients of the lags, p represents the number of lags taken from variable Y, and ϵ is an error term.

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \cdots + \beta_p Y_{t-p} + \epsilon_t$$

(3.3.1.1.1)

The equation (3.3.1.1.1) scales to multiple variables. For example, if you have two variables Y_1 and Y_2, to generate a forecast for Y_1, VAR leverages lags from Y_1 and Y_2. Similarly, to generate a forecast for Y_2, VAR leverages lags from Y_1 and Y_2. A simple system of equations to represent this is shown below, where $p = 1$.

$$Y_{1,t} = \alpha_1 + \beta_{11,1} Y_{1,t-1} + \beta_{12,1} Y_{2,t-1} + \epsilon_{1,t}$$
$$Y_{2,t} = \alpha_2 + \beta_{21,1} Y_{1,t-1} + \beta_{22,1} Y_{2,t-1} + \epsilon_{2,t}$$

(3.3.1.1.2)

3.3.1.2 Multiple Linear Regression

Multiple linear regression is simply the extension of simple linear regression [8]. It leverages multiple explanatory variables to forecast the outcome of one response variable. The simple linear regression general form equation is specified below, where y is the dependent variable, x is the independent variable, B is the slope, and A is the intercept.

$$y = Bx + A$$

(3.3.1.2.1)

There is only a slight difference between the simple linear regression equation and the multiple linear regression equation. The difference is that the equation for multiple linear regression considers multiple inputs rather than the single input in its univariate counterpart. The general form of multiple linear regression is shown below, where each subscript for $B_i x_i$ represents the separate independent variables taken into consideration.

$$y = B_1 x_1 + B_2 x_2 + \cdots + B_n x_n + A$$

(3.3.1.2.2)

3.3.2 Baseline Neural Networks

In this subsection, we will discuss baseline neural networks. This refers to the class of vanilla neural networks which are not modified in any way. The three primary categories of neural networks we will cover in this subsection can be grouped as feed-forward neural networks, convolutional neural networks, and recurrent neural networks.

Before detailing the specific neural network model architectures, the neural network optimization operation will be covered since all the neural network models are optimized using the same strategy. The neural network models are optimized by tuning their weights, trained over many training epochs. At each training epoch, the model's weights are optimized based on a loss function. To optimize the neural networks to best predict outputs based on the input data distribution, gradient descent is used to optimize model weights, which are the multiplicative factors of the weights in the feed-forward neural network (FFNN), the convolutional filters in the convolutional neural networks (CNN), and the weights in the hidden units of the recurrent neural networks (RNN). Each model weight is subject to the following at each training epoch (a subtraction of each weight via gradient descent, calculated by taking the negative gradient of the loss with respect to the model weights via backpropagation (chain-rule) [9]). In 5.2.1, θ represents the model weights, η is a learning rate, L is a loss function, and X are the model inputs.

$$\theta_{new} = \theta_{old} - \eta \frac{\partial L(X, \theta)}{\partial \theta} \tag{3.3.2.1}$$

3.3.2.1 Feed-Forward Neural Networks

The first type of baseline neural network used in this work is a simple feed-forward neural network [10]. The feed-forward neural network is the simplest and most fundamental class of artificial neural networks, consisting of an input layer, a hidden layer, and an output layer. Data moves in one direction, from input to output, and the hidden nodes in between the input and output layers adjust their weights based on training data and backpropagation to minimize error produced from a loss function. The feed forward neural network can be defined as such. Using input X, and L hidden layers (number of intermediate layers in a feed-forward neural network), this results in the following equations relating input X to target $\hat{y}$.

$$z_1 = W_1^T X + b_1$$

$$z_i = W_i^T a_{i-1} + b_i \quad \forall i = 2\ to\ L$$

$$a_i = f(z_i) \quad \forall i = 1\ to\ L$$

$$\hat{y} = W_{L+1}^T a_L + b_{L+1}$$

(3.3.3.2.1)

In 3.3.3.2.1, $\{(W_i, b_i)\}_1^{L+1}$ represents the set of weights and biases across all hidden and output layers of the feed forward network. f is the activation function used at the hidden layers which can be functions such as sigmoid or rectified linear unit (ReLU). For the problem of multivariate time series forecasting, this simple model unrolls the input matrix into a long vector.

3.3.2.2 Convolutional Neural Networks

CNNs are a class of neural networks initially conceived and pushed forward with great success in the field of computer vision [10]. They were inspired by the connectivity of neurons in the human brain and how these connections work regarding the visual cortex. Via the application of convolutional filters, a CNN can do well in capturing spatial and temporal dependencies within data. CNN models are a special type of neural network used for processing data with a known grid-like technology [10]. CNN models are typically used to analyze 1-D grid data such as time intervals and 2-D grid data such as images or pixel grids. CNNs utilize a mathematical operation called a convolution in at least one of its neural network layers which is a specialized linear operation.

Convolutions make use of kernels or filters (smaller matrices fitting into the input matrix with numerical values in each cell) which scan across input matrices by a specified stride length to produce feature maps (by multiplying corresponding and overlapping cells for both the filter and input matrix, summing these products, thus creating a sum of a linear combination between the filter and input matrix for each cell in a feature map). We can think of our input data for our problem application like image data, which are essentially pixel grids. Rather than pixels, each grid cell in our data is a variable for a bridge structure at a specific time step.

If we think of our data as a 2-D image, we utilize a 2-D kernel or filter to perform convolutions on our input samples. After convolution, feature maps are produced based on the

input image and a stack of filters, these feature maps are considered an embedding or understanding of the input matrix via the stack of filters. This is illustrated below in Figure 4.

Many filters are utilized so that different embeddings can be formed for the input matrix. The multiplicative factors in the filter cells act as weights in a model that uses convolutions. Typically, the feature maps produced from the filters are then flattened into vectors and connected to a simple feed forward neural network which also consist of weights and can take the embeddings generated from the convolutional layers and form a prediction.

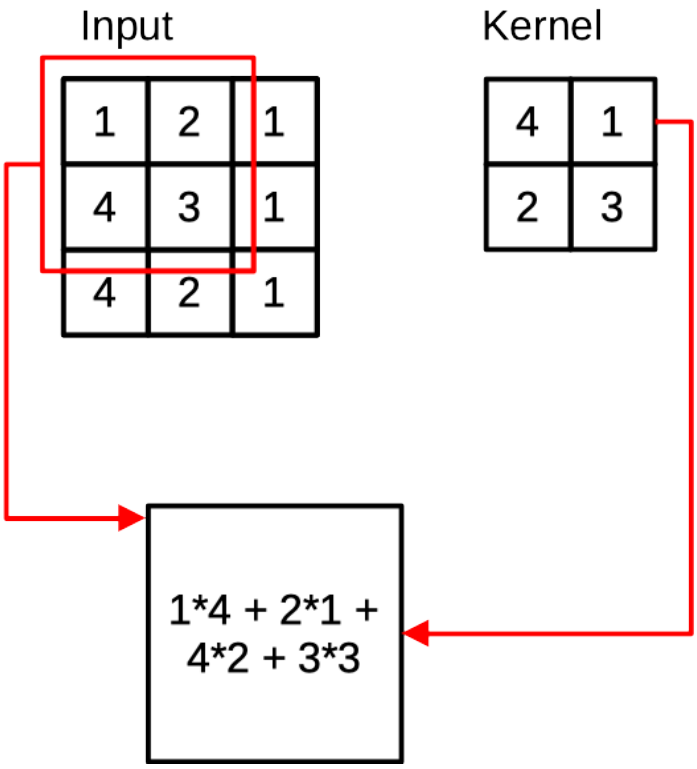

Figure 4: Illustration of the Convolution Operation

3.3.2.3 Recurrent Neural Networks

In this work, various types of RNNs were utilized. RNNs are a class of neural networks which have proven themselves to be useful for solving many sequence-based problems such as translation and text generation [10]. The reoccurring theme of all types of recurrent neural networks is that they consist of a hidden state which is shared across each step of a sequence. This hidden state act as the model's memory as it retains information from previous states. The primary difference between different types of RNNs is the formulation of the hidden state.

Below, is the formulation of a vanilla RNN, where activation function can be any activation function such as tanh. In addition, there is a figure illustrating the formulation in action. In Figure 5, *U*, *W*, and *V* are weights of the network, x_t is an input at time *t*, h_t is a hidden state at time *t*, and o_t is the output at time *t*. Note that RNNs can work for multivariate problems as the input and the weights corresponding to the input to the model interact via matrix multiplication.

$$a^{(t)} = b + Wh^{(t-1)} + Ux^{(t)}$$
$$h^{(t)} = activation\ function(a^{(t)})$$
$$o^{(t)} = Vh^{(t)}$$

(3.3.3.4.1)

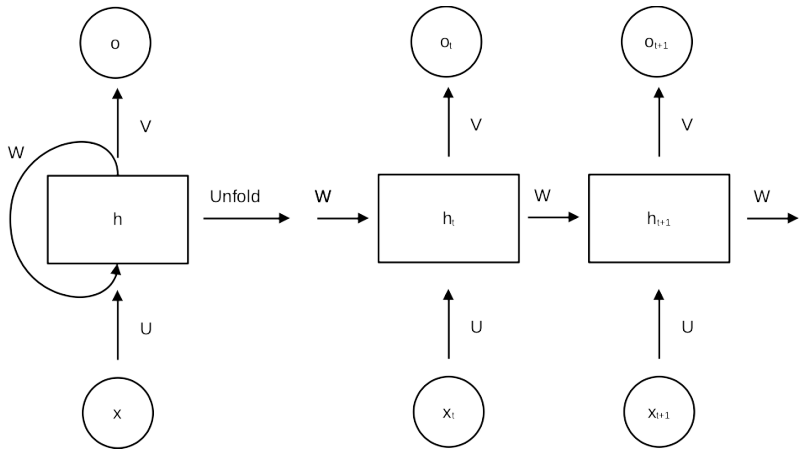

Figure 5: Illustration of RNN in Action

As mentioned previously, this work used different types of RNNs. The first type of RNN used was the Long Short-Term Memory (LSTM) architecture [11]; see Figure 6. LSTM networks are themselves RNNs with a more advanced hidden state which can better capture long-term dependencies within sequences. Their hidden states consist of both a hidden and cell state where flow of data is controlled via gating mechanisms which helps relinquish the issue of vanishing/exploding gradients present in vanilla RNNs during training time. The formulation for the hidden state of an LSTM is shown below as well as an illustration of the hidden unit. i_t represents the input gate of an LSTM at time step *t*, f_t is the forget gate of an LSTM at time step *t*, l_t is a set of candidate values to potentially be added to the hidden state, c_t is the cell state vector at time step *t*, o_t is the output gate of an LSTM at time step *t*, h_t is the hidden state output vector of an LSTM at time step *t*, and the *W* matrices are model weight specific to each gate.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$
$$l_t = tanh(W_l \cdot [h_{t-1}, x_t] + b_l)$$
$$c_t = f_t \cdot c_{t-1} + i_t \cdot l_t$$
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$
$$h_t = o_t \cdot \tanh(c_t)$$

(3.3.3.4.1)

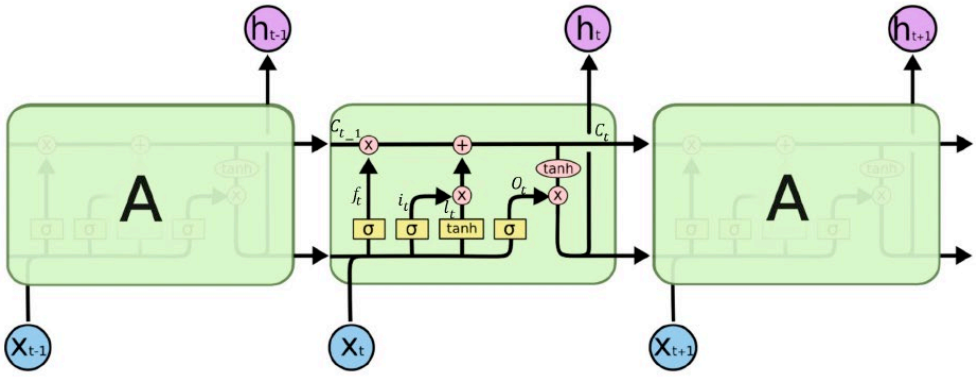

Figure 6: LSTM Hidden State Unit *[11]*

Another RNN based architecture used in this work is a Gated Recurrent Unit (GRU) network [11]; see Figure 7. This architecture is like LSTM networks in that they are RNNs with complex hidden state cells which have been similarly designed to handle long-term sequence dependencies better via gating mechanisms. The gates that are utilized in GRU networks are update and reset gates which determine which information should be carried forward at each time step. The formulation of the GRU hidden state as well as an illustration showing this is provided below. The GRU forms a new update gate by integrating the forget and input gates.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$$
$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$$
$$\tilde{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t])$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

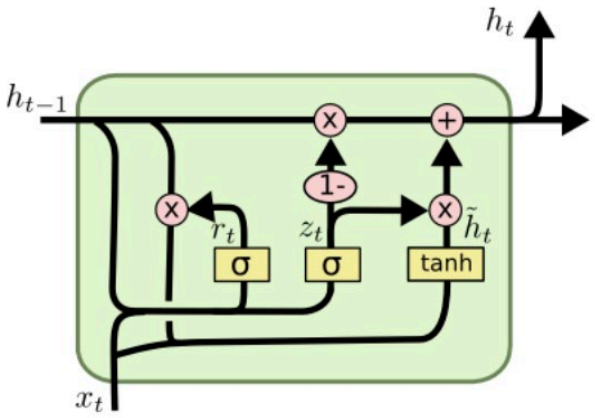

Figure 7: GRU Hidden State Unit *[11]*

The final RNN based model used in this work is the Bidirectional LSTM (BiLSTM) network [12]. This model combines two independent LSTM networks together to obtain both backward and forward information about a sequence at every time step. While using BiLSTM networks, the model will examine time series input both from past to future and from future to past.

3.3.3 Novel Neural Networks

In this subsection, we will discuss novel neural networks. This refers to the class of neural networks which are modified further than the baseline.

3.3.3.1 Temporal Convolutional Network

The Temporal Convolutional Network (TCN) is a 1-D fully convolutional network which leverages causal and dilated convolutions to successfully obtain long term memory for time series data [13]. This network has proven to exhibit longer term memory than RNN based architectures while also offering the advantage of a more stable and efficient model. This architecture is entirely convolutional and can provide time series output for variable length inputs. The TCN uses a 1-D fully convolutional network architecture [14] where each hidden layer is the same length as the input layer and zero padding of length (kernel size – 1) is added to keep subsequent layers the same length as prior layers.

Causal convolutions indicates that there can be no information leakage in the model from the future into the past. The TCN accomplishes this by ensuring that outputs at time *t* are convolved using only elements from time *t* and earlier. Zero padding is utilized to ensure that

elements at the very beginning of a sequence still have values to convolve over causally. In addition, dilated convolutions are used in the TCN. Dilation refers to expanding the receptive field of a convolution operation by some factor, effectively adding distance between elements of an input sequence utilized when computing entries of an output sequence. This allows the model to have an exponentially larger receptive field. Via a dilation factor, the receptive field of the TCN can be increased, this is illustrated in Figure 8.

The actual convolutional operation is shown in Figure 9 where a 1-D kernel is applied to a 1-D input. Note that this can be extended further to multivariate sequence forecasting as shown in Figure 10 by increasing the number of input channels and increasing the number of kernel input channels.

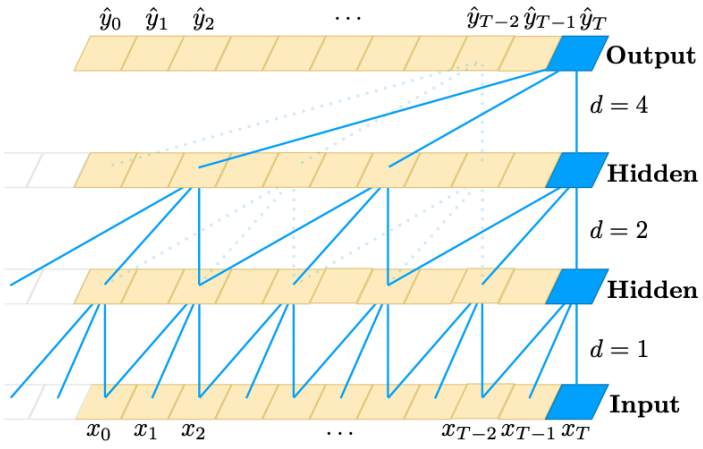

Figure 8: Causal and Dilated Convolutions in TCN where *d* is the Dilation Factor *[13]*

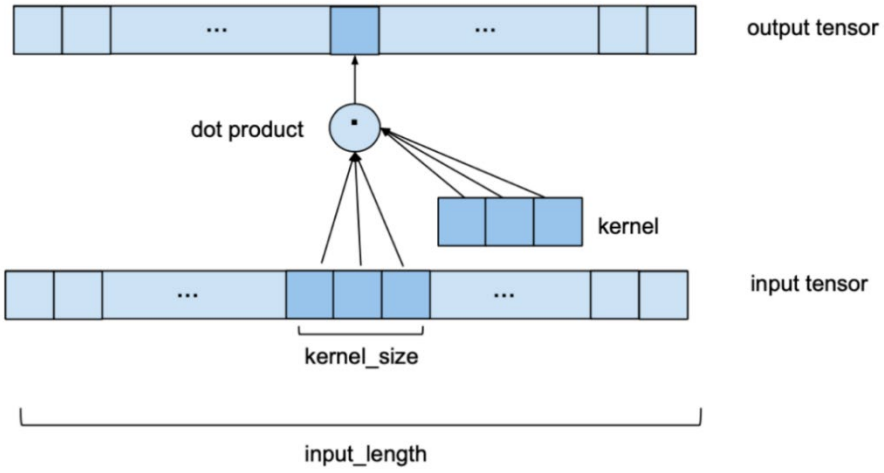

Figure 9: Convolution Operation for TCN *[15]*

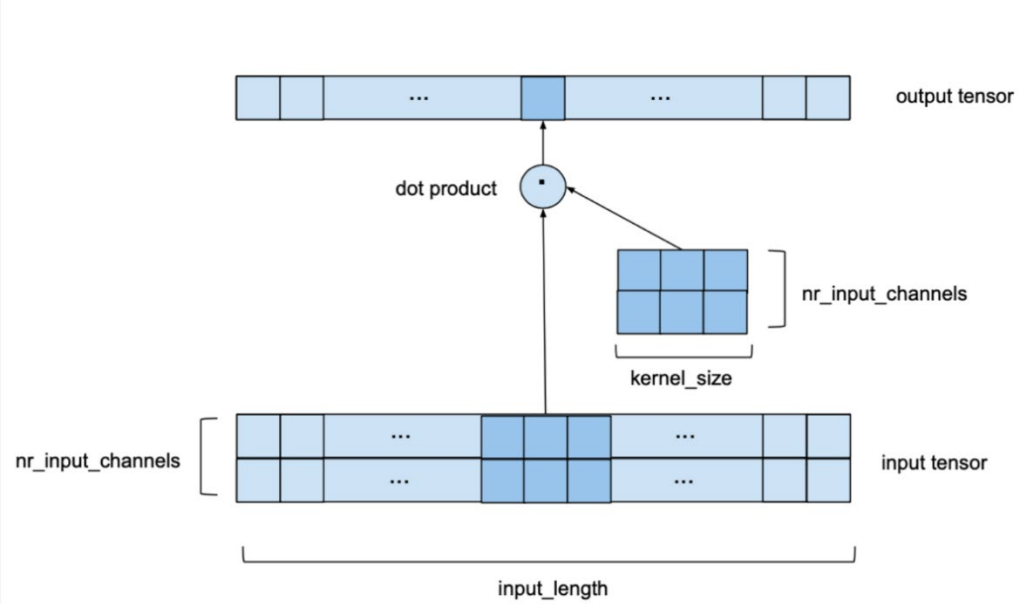

Figure 10: Multivariate Convolution Operation for TCN

3.3.3.2 Multi-Channel CNN

The Multi-Channel CNN is a novel CNN architecture formulated specifically with multivariate time series classification in mind [16]. The intuition behind this model is that it takes as input a multivariate time series which has been separated into individual univariate sequences. Several individual CNNs of identical structure are then utilized to obtain a representation of each univariate time series. The final layers of this architecture consist of a concatenation of representation vectors followed by a few feed-forward layers which produce a classification or forecasting output. This architecture is illustrated in Figure 11.

3.3.3.3 CNN + BiLSTM

The CNN + BiLSTM model is simply a CNN with a BiLSTM attached at the end of it. The reasoning behind this model architecture is to utilize the CNN to learn a better representation of the initial time series and then further use the BiLSTM to forecast on this new latent space.

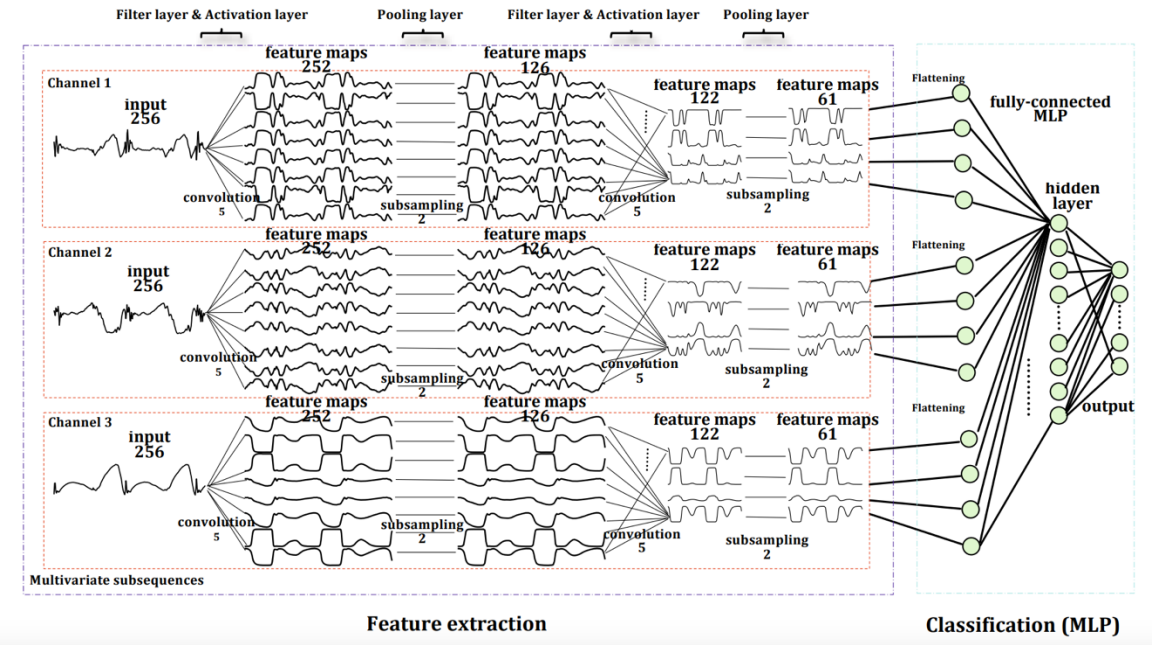

Figure 11: Multi-Channel CNN Architecture *[16]*

3.4 Experimental Evaluation

In this section we will go into experimental evaluation of the models specified above, how they were used in this specific problem, and how the added complexity when moving from traditional regression-based models to novel neural networks did indeed allow for better performance.

We performed experiments using solely NBI data as well as NBI data supplemented with NOAA data. The experiments consisted of raw as well as normalized data for the purpose of training our models. A 60:40 train test split was utilized to evaluate the performance of our models with the root mean squared error (RMSE) metric. The normalization method used was min-max scaling. The models were conditioned on multi-variate time series data and forecasted future condition ratings for Deck, Substructure, and Superstructure. An RTX 2070 Super was leveraged as hardware acceleration to parallelize the training of our neural networks. Tensorflow 2.5.0 was utilized to develop our models.

RMSE was used to measure the error in the produced outputs during evaluation of the test split of the data after model training was complete. RMSE was measured for each individual condition rating outputted by the model. The individual RMSE values are calculated for all

samples in the test split then averaged to obtain an aggregate measure. The lower this value is, the better.

Figure 12 is a plot of loss over time as we trained a TCN as an example to show how model training was monitored. We also provide an example of predictions overlayed on top of ground truth data points in Figure 13.

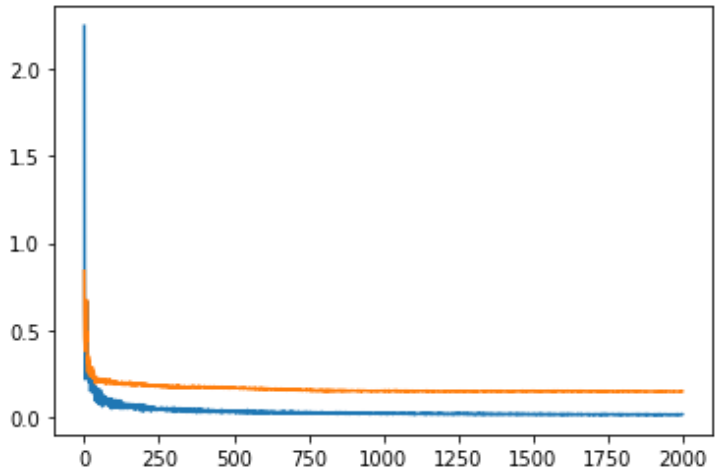

Figure 12: Sample Training Plot for TCN Model. Blue Represents Training Loss Over Epochs and Orange Represents Validation Loss Over Epochs

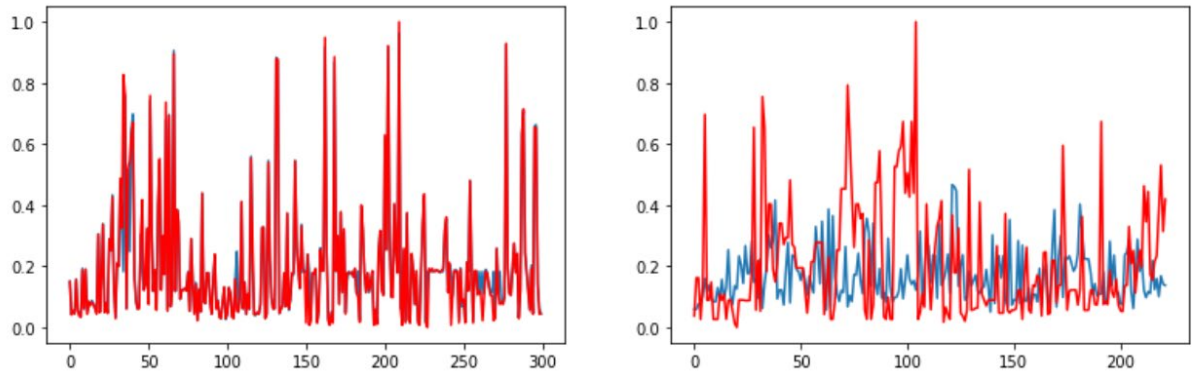

Figure 13: Sample Forecasting Plots for TCN Model. On the Left is Forecasting for the Last Time Step of Individual Bridge Time Series from Training Data and on the Right is Forecasting for the Last Time Step of Individual Bridge Time Series from Test Data

In the rest of this section, we present our experimental results. We begin by showing our results from the traditional regression-based models selected below (Tables 3 to 5).

	Score for Deck Condition Rating on Raw Dataset	Score for SupStruct Condition Rating on Raw Dataset	Score for SubStruct Condition Rating on Raw Dataset	Score for Deck Condition Rating on Normalized Dataset	Score for SupStruct Condition Rating on Normalized Dataset	Score for SubStruct Condition Rating on Normalized Dataset
Vector Autoregression NBI Only RMSE	3.05	4.18	3.18	5.61	5.75	5.53
Vector Autoregression NBI + NOAA RMSE	3.53	4.95	0.64	5.75	5.77	1.69

Table 3: RMSE Results for Vector Autoregression

	Score for Deck Condition Rating on Raw Dataset	Score for SupStruct Condition Rating on Raw Dataset	Score for SubStruct Condition Rating on Raw Dataset	Score for Deck Condition Rating on Normalized Dataset	Score for SupStruct Condition Rating on Normalized Dataset	Score for SubStruct Condition Rating on Normalized Dataset
Multiple Regression NBI Only RMSE	0.29	0.27	0.26	0.29	0.27	0.26
Multiple Regression NBI + NOAA RMSE	1.01	0.90	0.92	1.02	0.91	0.92

Table 4: RMSE Results for Multiple Regression

	Score for Deck Condition Rating on Raw Dataset	Score for SupStruct Condition Rating on Raw Dataset	Score for SubStruct Condition Rating on Raw Dataset	Score for Deck Condition Rating on Normalized Dataset	Score for SupStruct Condition Rating on Normalized Dataset	Score for SubStruct Condition Rating on Normalized Dataset
Double Exponential Smoothing RMSE	0.10	0.09	0.09	0.10	0.09	0.09

Table 5: RMSE Results for Double Exponential Smoothing

Note that for the double exponential smoothing model, it is entirely univariate, so it can only forecast condition ratings solely on their history rather than take in additional information from other variables. In addition to this, all the traditional regression-based models can only obtain predictions for samples one at a time. This means that they are not able to obtain a comprehensive view of all samples at once to decide about predictions.

In contrast, the neural network models can handle the multi-variate nature of the data as well as be trained on the entirety of the sample set to gain a more wholistic view of the data to make better informed and well-rounded decisions. Below, we display the results from the neural network models (Tables 6 and 7).

	Score for Deck Condition Rating on Raw Dataset	Score for SupStruct Condition Rating on Raw Dataset	Score for SubStruct Condition Rating on Raw Dataset	Score for Deck Condition Rating on Normalized Dataset	Score for SupStruct Condition Rating on Normalized Dataset	Score for SubStruct Condition Rating on Normalized Dataset
Linear Model RMSE	0.34	0.33	0.29	0.93	0.77	0.92
CNN Model RMSE	0.32	0.18	0.25	0.88	0.65	0.81
LSTM Model RMSE	0.34	0.23	0.27	1.88	1.86	1.91
BiLSTM Model RMSE	0.38	0.24	0.25	0.91	0.72	0.79
GRU Model RMSE	0.33	0.21	0.24	8.08	8.22	8.05
TCN Model RMSE	0.32	0.23	0.24	1.74	1.76	1.81
CNN + BiLSTM Model RMSE	0.57	0.34	0.40	0.82	0.68	0.75
Multi-Channel CNN Model RMSE	0.64	0.43	0.55	0.83	0.83	1.03

Table 6: RMSE Results for Neural Networks on NBI only

	Score for Deck Condition Rating on Raw Dataset	Score for SupStruct Condition Rating on Raw Dataset	Score for SubStruct Condition Rating on Raw Dataset	Score for Deck Condition Rating on Normalized Dataset	Score for SupStruct Condition Rating on Normalized Dataset	Score for SubStruct Condition Rating on Normalized Dataset
Linear Model RMSE	0.33	0.19	0.26	0.97	0.77	0.93
CNN Model RMSE	0.32	0.18	0.26	0.90	0.75	0.87
LSTM Model RMSE	0.37	0.21	0.25	1.82	1.79	1.89
BiLSTM Model RMSE	0.35	0.21	0.27	1.92	1.93	1.95
GRU Model RMSE	0.34	0.20	0.24	9.52	9.74	9.55
TCN Model RMSE	0.36	0.26	0.25	1.91	1.96	2.00
CNN + BiLSTM Model RMSE	0.51	0.30	0.36	0.79	0.59	0.84
Multi-Channel CNN Model RMSE	0.66	0.43	0.53	1.31	1.18	0.81

Table 7: RMSE Results for Neural Networks on NBI and NOAA data

Based on the results for the models shown, the TCN and CNN models consistently perform the best compared to the other models. The case of the double exponential smoothing model having a lower RMSE than the other models is not representative of its performance since it is entirely univariate, and we are examining a multi-variate problem. The success of the CNN and TCN models can potentially be attributed to them being convolutional in nature and the input into these models is a matrix with dimensions dependent on the number of time steps as well as the number of variables. This matrix-like input into the models may allow them to derive a better understanding or embedding of the input data when compared against the other type of neural networks. It was expected that the neural networks would perform better than the traditional regression-based models as they provide an added complexity which proved to be crucial in improving performance.

To reiterate, as we had hoped for during our research phase, the CNN and TCN models outperformed our other methods. The TCN architecture captures a wider context of our time series with its use of dilated convolutions which was a key factor in its success.

We believe our neural network models performed much better than the regression-based models as the neural network models were able to effectively capture a holistic view of the entire dataset, meaning, that they were optimized over all the separate bridge time series, thus providing a more informed model. In contrast, the regression-based approach required that an individual regression model be generated for each separate bridge time series and in the end the RMSE score for the regression-based models was derived by averaging out all the separate RMSE scores across each of the individual bridge time series models, this approach severely limits the amount of data each model can view, thus resulting in a poor overall understanding of the dataset with this method. Note that the neural network-based models can also be trained further if more data were to be collected while the regression-based approaches only can learn from a single time series at a time.

3.5 Tool Description and Sample Results

In addition to the modeling experiments and results for this work, an end-to-end system was also developed for bridge engineers to interface with to obtain forecasts for future years of condition ratings. In addition to this, the system also automates the process of retrieving and integrating new relevant data for the models to be retrained so that their inferences are more up to date. Figure 14 is a high-level view of the initial system design.

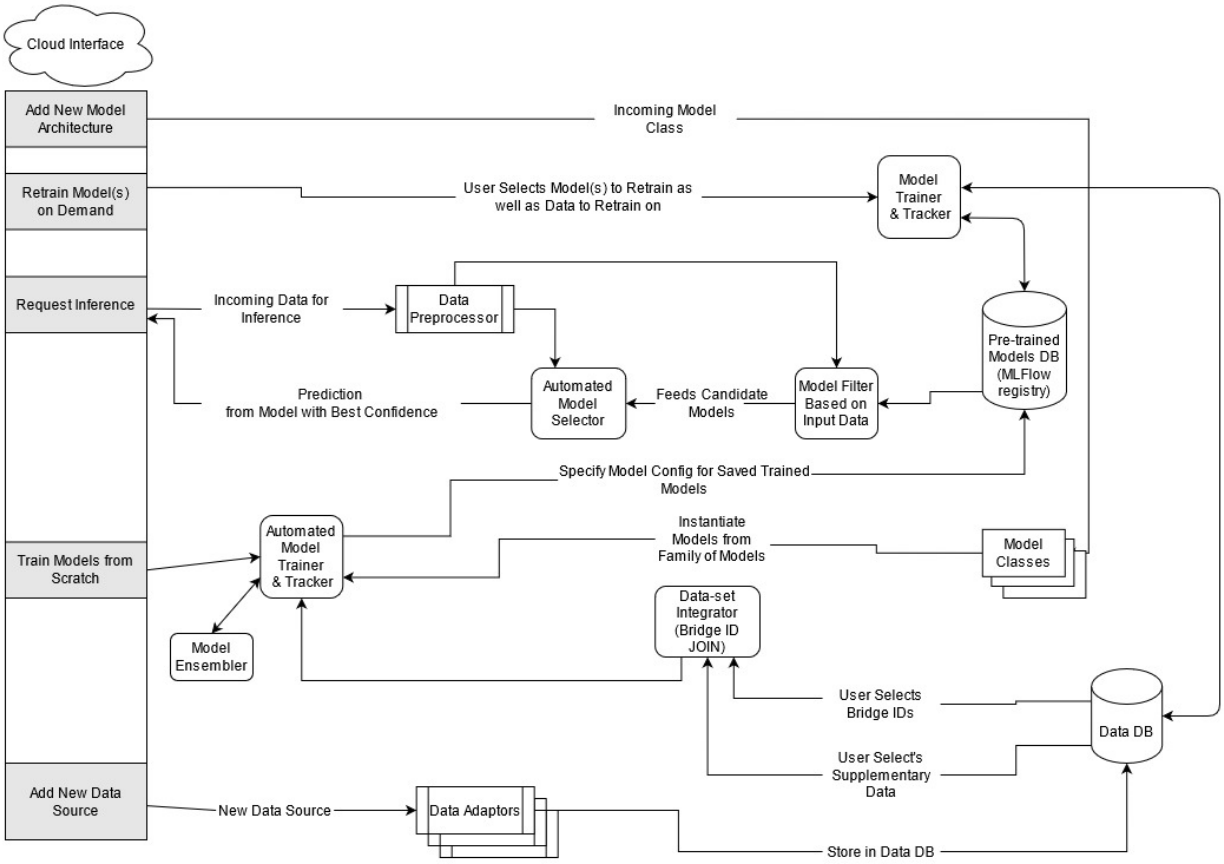

Figure 14: Bridge Management as a Service Software Tool Diagram

There are several key components of the software system. The high-level interfacing modules exposed to the user in the product for bridge deterioration forecasting are kept as simple as possible for the user while still enabling features that empower a bridge engineer to make more informed decisions about bridge structures.

The features for bridge deterioration forecasting can be split into two primary categories: (1) Forecasting and (2) Model retraining.

For forecasting, this is simply the utilization of the highest performing model in the system to obtain a forecast for all bridge samples in the dataset for the next N years where N is determined by the system user. The models in the system are managed via MLFlow, meaning, the storage, organization, and utilization of specific models in the system are all automated via a MLFlow registry allowing for models to be pulled and used in an intelligent manner such that only models that are high performing are utilized. An inference through the system is as easy as

providing the number of years to predict and selecting a model trained on a specific dataset combination. This is illustrated Figure 15 below.

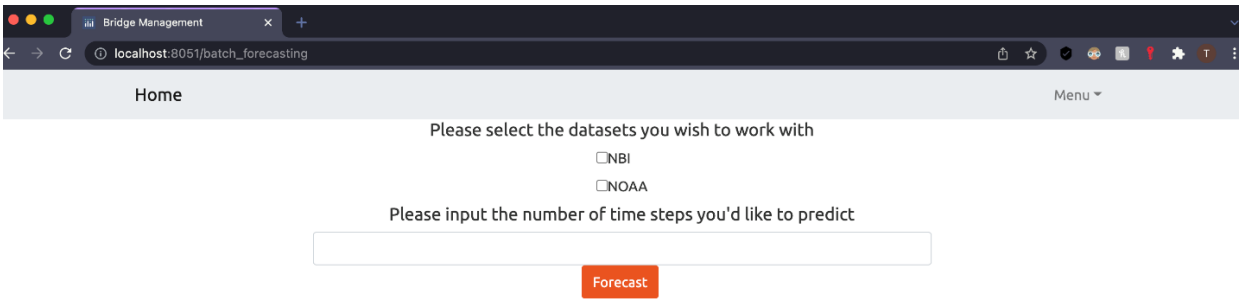

Figure 15: System Interface for Forecast Inference

The resulting output of the forecast inference is a .zip file containing an Excel file for each condition rating where each file contains predictions for each bridge sample in the dataset. The columns included in the output Excel files are, in order, Bridge ID, Condition Rating Type, Input Time Steps, and Prediction Time Steps. This is illustrated in Figure 16 below.

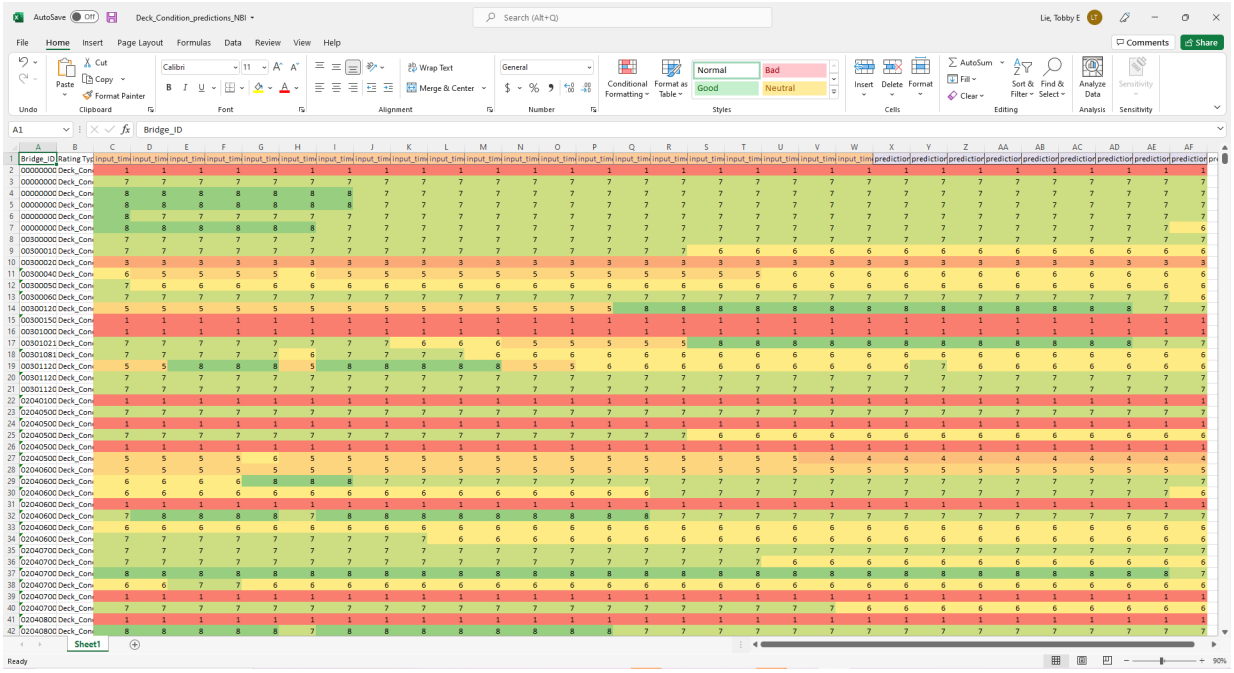

Figure 16: Example Output from System Forecast Enhanced using Color Schemes

For model retraining, there are many functions which are automated to enable bridge engineers to have models that are consistently updated based on newly available data, whether that be annual NBI or NOAA data. A user can select models to be retrained based on a dataset combination, the system will automatically scrape new data from the corresponding datasets, adapt them to the proper format, and integrate them into the system for the models to be retrained on. Again, MLFlow is used to track and maintain the best performing models for the given dataset combination. At the end of retraining, the best performing models will be persisted in the system to be used for inference in the forecasting module. There are also components in place that automate the process of preprocessing and integrating new data into the system. In addition, there is an option to train models on specific clusters of bridge samples which are generated from a different part of the system known as bridge family generation, which is covered in a following section. There is also a way for users to specify if they would like bridges to be trained on non-repaired bridges only, meaning, all bridges that exhibit some sort of increase in condition rating at some point in the input data are removed for training. If a user would like to train their model using NOAA data, the data scraper requires a unique NOAA API token which can be obtained via a unique email on the NOAA website. The goal of the system is to ensure user convenience. The retrain module interface is shown in Figure 17 below.

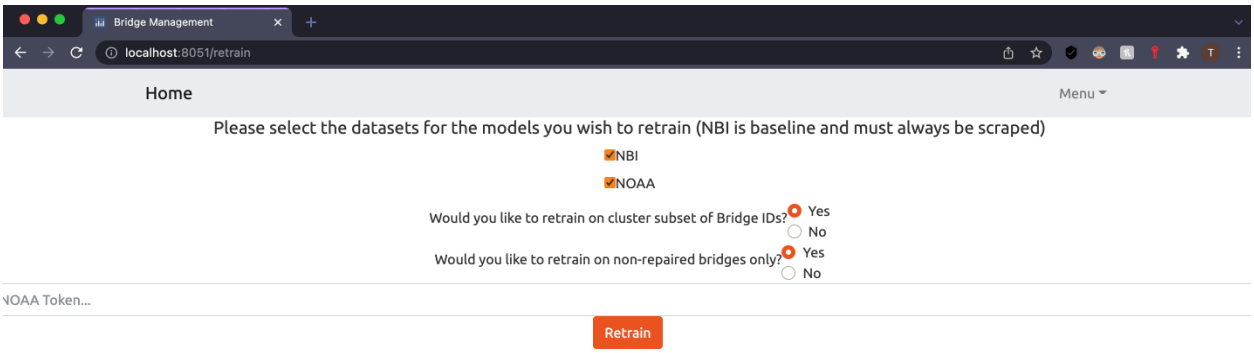

Figure 17: System Interface for Model Retraining

As a final note, we again echo the sentiment that the goal of this work was not only to provide experimental results for the models we explored, but to also provide a tangible product. This product was developed with the intention of making it as easy as possible for bridge

engineers to obtain predictions. Again, the key focus of this line of work is to make decisions more intelligently about resource allocation for bridge assets. The hope is that this tool will enable bridge engineers to better inform their processes. Usage of the tool requires no prior knowledge on the software side and enables easy access to high performing models on relevant data to help make important decisions.

4. Bridge Family Generation (Subtyping)

4.1 Problem Definition

Bridge family generation (aka bridge subtyping or bridge performance clustering) is the process of grouping a set of bridges into subtypes, where within each subtype/subgroup, bridge performances are similar, while bridge performance across different subtypes/subgroups is largely different. Bridge subtyping has numerous applications, including the following two use-cases. On the one hand, by subtyping bridges, bridge engineers can identify different performance behaviors observed in the set of bridges under their supervision, and focus on handling bridges with inferior performance. Moreover, they can study each subgroup separately to investigate the roots of their performance shortcomings. On the other hand, bridge engineers can apply bridge deterioration forecasting models (such as those presented in Section 3) to bridge subgroups (rather thank the entire set of bridges) for improved accuracy in forecast.

For effective bridge subtyping, one needs to devise efficient solutions for (1) data preprocessing, (2) accurate time series data clustering, and finally, (3) effective interpretation of the subtypes to adjust the clustering model. In particular, we have developed and evaluated a variety of different clustering models, including a novel semi-supervised clustering model that outperforms other existing methods.

4.2 Literature Review

In this section, we will go over the literature review for Bridge Subtyping with a focus on Bridge Descriptive Analysis (Clustering). Chetti et al [17] proposed an approach for producing clusters of bridges with similar features associated with sufficiency ratings over user-defined periods of time via correlation network modeling as well as Markov clustering. The bridges they applied this methodology for were steel bridges with stringer/multi-beam or girder designs. After they extracted the top five clusters based on a clustering coefficient, they performed two types of analysis: Analysis of bridge behavior with respect to sufficiency rating as well as analysis of the top five clusters with respect to input rating parameters. In the first analysis, they elected two bridges from the fifth cluster in their top five clusters to look at their behavior in terms of their overall sufficiency ratings. In the second analysis performed on the top five clusters, various input rating parameters of output ratings, such as sufficiency rating were considered for cluster

enrichment analysis. They compared the top 5 clusters' average input ratings such as Deck rating, Superstructure Rating, Substructure Rating, Water Adequacy Rating, and Structural Condition Rating. They plotted distributions of ratings for each cluster and analyzed how enriched distributions of ratings were, regarding rating thresholds that deem a bridge as structurally deficient in order to get a sense of input ratings that directly correlated to low or high sufficiency ratings. By forming this correlation to specific input parameters, they concluded that experts could examine closely which ratings were causing a low sufficiency rating.

Chang et al [18] performed bridge clustering on massive datasets of bridges with the goal of executing systematic recognition of damage patterns on bridge elements. The primary objective of this research was to group bridges with similar characteristics so that damage patterns for bridge elements could be recognized precisely. To evaluate the validity of prestressed concrete I-shaped bridge clustering, damage pattern recognition on bridge elements for each cluster was investigated in two steps. Based on 12,980 inspection records, 5 clusters were formed. In the first step, the relative occurrence frequency of damage to the five target elements -- expansion joint, pavement, deck, girder, and cross beam -- based on the five clusters was visualized graphically. By performing this first step, the research team was able to quantitatively analyze the difference in the damage that occurred in conjunction with the bridge characteristics of each cluster derived from the clustering results. In the second step, the research team utilized logistic regression analysis to quantitatively investigate the effects of the five characteristics selected as bridge clustering features on the occurrence of element damage found in each cluster.

Diez et al [19] presented an approach that combined feature extraction via a nearest neighbor-based outlier removal, followed by a clustering approach over both vibration events and joint representatives. Vibration signals triggered by vehicles passing on a bridge from different joints were classified so that damaged joints could be detected and located. Based on data the research team collected via fitting the Sydney Harbour Bridge with sensors, they were able to validate their clustering results by applying an iterative KNN (K Nearest Neighbors) based outlier, noisy signal removal, and a Fourier transform of resulting joint events. Then K-means based clustering of both joint events and joint representatives was performed. Their clustering results indicated similarity between joints located in different bridge locations which helped to group joints with similar behavior.

Knight et al [20] provided an analysis focused on identifying distinct and important subgroups of bridges. A representative sample of bridge structures from this method could be used for long-term monitoring to learn about the degradation processes and how these processes interact with different bridge structures, material types, and other variables associated with bridge performance. The mean values for each of the continuous variables utilized in the analysis of the clustering were provided in a table format. For example, the bridges identified in cluster 1 had a mean age of 21.7 years and a mean average daily traffic (ADT) of 7.9 percent. They argued that many observations could be made by investigating results like the fact that clusters 1-7 most likely contain only bridges that had not been reconstructed during their service lives whereas clusters 8-11 contained bridges that were and were not reconstructed during their service lives. These types of comparisons could help to show the differences between the different clusters and provide support for their practical significance. Similarly, the properties of indicator variables for each cluster were provided. The information in this context indicates the probability of a bridge having one of the two levels of each indicator. For example, all bridges included in cluster 1 had a design load other than DL2. Like the continuous data, much information could be obtained from this type of analysis concerning the properties of each cluster. This analysis showed how each cluster was related to a specific portion of the original sample and provided insight that was not evident from an initial visual inspection.

4.3 Proposed Methods

In this section, we present a number of clustering models we have developed and utilized to address the bridge subtyping problem.

4.3.1 Time Series Learn

Temporal data are ubiquitous in many application domains, such as medicine and robotics. Dealing with such data requires methods that take into account the high correlation between consecutive samples in a time series. Moreover, in many cases, one would like a time series approach to encode invariance to small time shifts, which once again implies using specific methodologies. Time Series Learn (or TS Learn for short) [21] is a Python package that offers a variety of classical models for clustering time series data, such as bridge performance data. In particular, we have utilized this package to evaluate performance of the following classical methods for time series clustering: KMeans model (including the variations of K-

Means with Euclidean distance measure (*KMeans Euclidean*), KMeans with Dynamic Time Warping (DTW) distance measure (*KMeans DTW*), KMeans with Soft Dynamic Time Warping (SoftDTW) distance measure (*KMeans SoftDTW*), and Kernel KMeans (*Kernel K-Means*)), and K-shape model. For more details about these models, we refer the reader to description of the TS Learn package [21]. The code that implements these methods is available here: https://github.com/tslearn-team/tslearn

4.3.2 Deep Temporal Clustering (DTC)

Deep Temporal Clustering (DTC) [22] is a clustering model which naturally integrates dimensionality reduction and temporal clustering into a single end-to-end learning framework, fully unsupervised. The algorithm utilizes an autoencoder for temporal dimensionality reduction and a novel temporal clustering layer for cluster assignment. Then it jointly optimizes the clustering objective (using KMeans algorithm also mentioned above) and the dimensionality reduction objective. In particular, we have developed a number of variations of DTC based on the type of distance measure used for object clustering, namely: DTC-KMeans with Euclidean Distance (*DTC-Kmeans-eucl*), DTC-KMeans with Correlation Coefficient-based Distance (*DTC-Kmeans-cor*), and DTC-KMeans with Complexity-Invariant Distance (*DTC-Kmeans-cid*). The code that implements these methods is available here: https://github.com/FlorentF9/DeepTemporalClustering

4.3.3 COBRASTS

COBRASTS [23] is a semi-supervised time series clustering model. This novel technique requires user input to link and unlink the relationship between samples of time series data so that COBRASTS can identify clusters that are characterized by small local patterns. A small amount of semi-supervision can greatly improve clustering quality for time series; the choice of the clustering algorithm matters. It is possible to customize this clustering algorithm and similarity metric based on the dataset to optimize the clustering result. In particular, we have developed a number of COBRASTS customizations as follows: COBRASTS with KMeans and Euclidean Distance (*COBRASTS-Kmeans-eucl*), COBRASTS with KMeans and Euclidean Distance (*COBRASTS-Kmeans-dtw*), and COBRASTS with KMeans and Dynamic Time Warping Distance (*COBRASTS-Kshapes*). The code that implements these methods is available here:

https://github.com/ML-KULeuven/cobras

4.3.1 Clustering by Semi-Supervised Learning (SSL)

Semi-supervised Embedding for Scalable and Accurate [23] is our proposed semi-supervised learning (SSL) clustering framework. This novel framework relied on scalable and accurate autoencoder-based semi-supervised learning for time series clustering in the embedded space. This model leverages a small subset of labeled examples to significantly improving the quality of the autoencoder's learned latent space for clustering. We have implemented a number of variations for our SSL based solution, with or without Autoencoder (AE) embedding, and in each case with a variety of distance measures. In particular the following variations of the SSL method are developed and evaluated: SSL with Protocol distance measure (*SSL-proto*), SSL with Database distance measure (*SSL-db*), SSL with Silhouette distance measure (*SSL-silh*), SSL without autoencoder without any distance measure (*SSL-ae*), SSL with autoencoder with Protocol distance measure (*SSL-ae-proto*), SSL with autoencoder with Database distance measure (*SSL-ae-db*), and SSL with autoencoder with Silhouette distance measure (*SSL-ae-silh*). The code that implements these methods is available here: https://github.com/nnbaokhang/Semi_Supervised_Embedding_for_Scalable_and_Accurate

4.4 Experimental Evaluation

We performed the experimentation using the NBI dataset presented in Section 2. Specifically, we used Sufficiency Rating, Deck Condition Rating, Super Structure Condition Rating, and Sub Structure Condition Rating for bridge performance family generation. We performed data preprocessing to convert these features to time-series data. To evaluate the performance of the proposed methods, we used the Silhouette Coefficient. As shown in Figure 18, Silhouette Coefficient is calculated using the mean intra-cluster distance (a) and the mean nearest-cluster distance (b) for each sample. The Silhouette Coefficient for a sample is then computed as $(b - a) / max(a, b)$. To clarify, here b is the distance between a sample and the nearest cluster that the sample is not a part of. The larger the silhouette coefficient, the better the quality of the generated subtypes (i.e., tighter clusters as well as more separation between clusters), with value 1 the highest value of the coefficient.

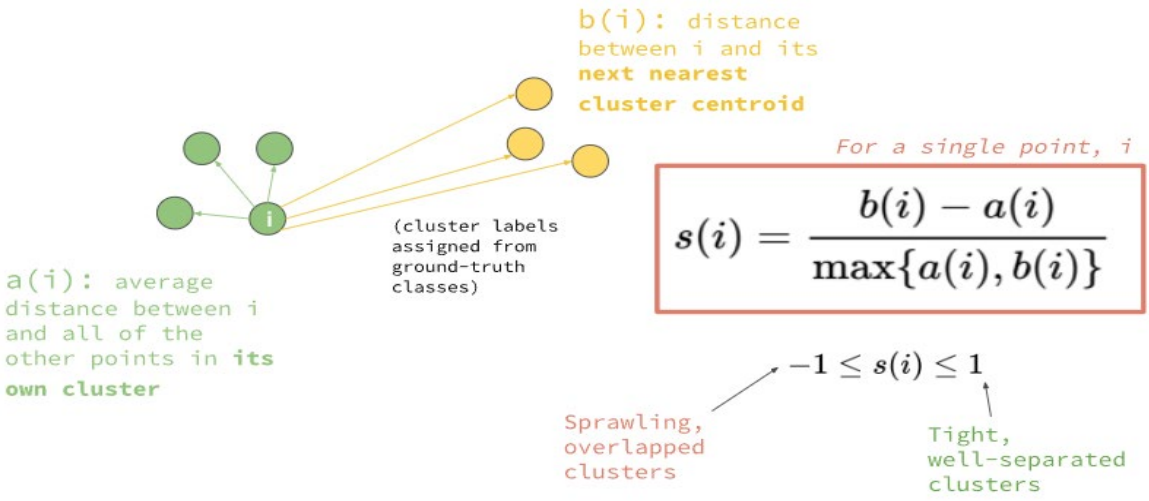

Figure 18: Silhouette Coefficient for evaluation of subtyping methods

Our evaluation results are shown below, in Tables 8-11, corresponding to the 4 aforementioned features, respectively. As shown in the tables, in all cases, some variation of our proposal SSL method has the maximum silhouette coefficient and therefore, outperforms other existing methods for bridge family generation.

	Normalized Sufficiency Rating
SSL-ae-sihl	0.894
SSL-ae-db	0.63719
SSL-db	0.6329
SSL-silh	0.5107
Cobrasts k-means-euclidean	0.4833
SSL-ae-proto	0.4569
SSL-ae	0.448
SSL-proto	0.331
K means Softdtw	0.2844
Kernel k-means	0.2245
K Means Euclidian	0.22
DTC-k-mean-eucl	0.196
DTC-k-mean-cor	0.19
K-Shape	0.1878
K-means DTW	0.187
DTC-k-mean-cid	0.134
Cobrasts k-shapes	0.1045
Cobrasts k-means-dtw	-0.7503

Table 8: The Silhouette Loss Function Results for Normalized Sufficiency Rating

	Normalized Deck Condition Rating
SSL-ae-sihl	0.338
SSL-ae-db	0.592
SSL-db	0.779
SSL-silh	0.884
Cobrasts k-means-eucliedean	0.429
SSL-ae-proto	0.3
K means Softdtw	0.595
Kernel k-means	0.597
K Means Euclidian	0.591
DTC-k-mean-eucl	0.561
DTC-k-mean-cor	0.592
K-Shape	0.503
K-means DTW	0.598
DTC-k-mean-cid	0.525
Cobrasts k-shapes	-0.153
Cobrasts k-means-dtw	-0.069

Table 9: The Silhouette Loss Function Results for Normalized Deck Condition Rating

	Normalized Super Structure Condition Rating
SSL-ae-sihl	0.804
SSL-ae-db	0.67
SSL-db	0.375
SSL-silh	0.737
Cobrasts k-means-eucliedean	0.48
SSL-ae-proto	0.368
K means Softdtw	0.63
Kernel k-means	0.591
K Means Euclidian	0.595
DTC-k-mean-eucl	0.5149
DTC-k-mean-cor	0.592
K-Shape	0.503
K-means DTW	0.553
DTC-k-mean-cid	0.56
Cobrasts k-shapes	0.3336
Cobrasts k-means-dtw	-0.167

Table 10: The Silhouette Loss Function Results for Normalized Super Structure Condition Rating

	Normalized Sub Structure Condition Rating
SSL-ae-sihl	0.747
SSL-ae-db	0.4381
SSL-db	0.3645
SSL-silh	0.7367
Cobrasts k-means-eucliedean	0.5784
SSL-ae-proto	0.353
K means Softdtw	0.6328
Kernel k-means	0.6279
K Means Euclidian	0.5536
DTC-k-mean-eucl	0.575
DTC-k-mean-cor	0.5905
K-Shape	0.501
K-means DTW	0.5812
DTC-k-mean-cid	0.596
Cobrasts k-shapes	-0.1357
Cobrasts k-means-dtw	0.787

Table 11: The Silhouette Loss Function Results for Normalized Super Structure Condition Rating

4.5 Tool Description and Sample Results

Figure 19 shows the procedure to use the bridge family generation tool to create bridge families.

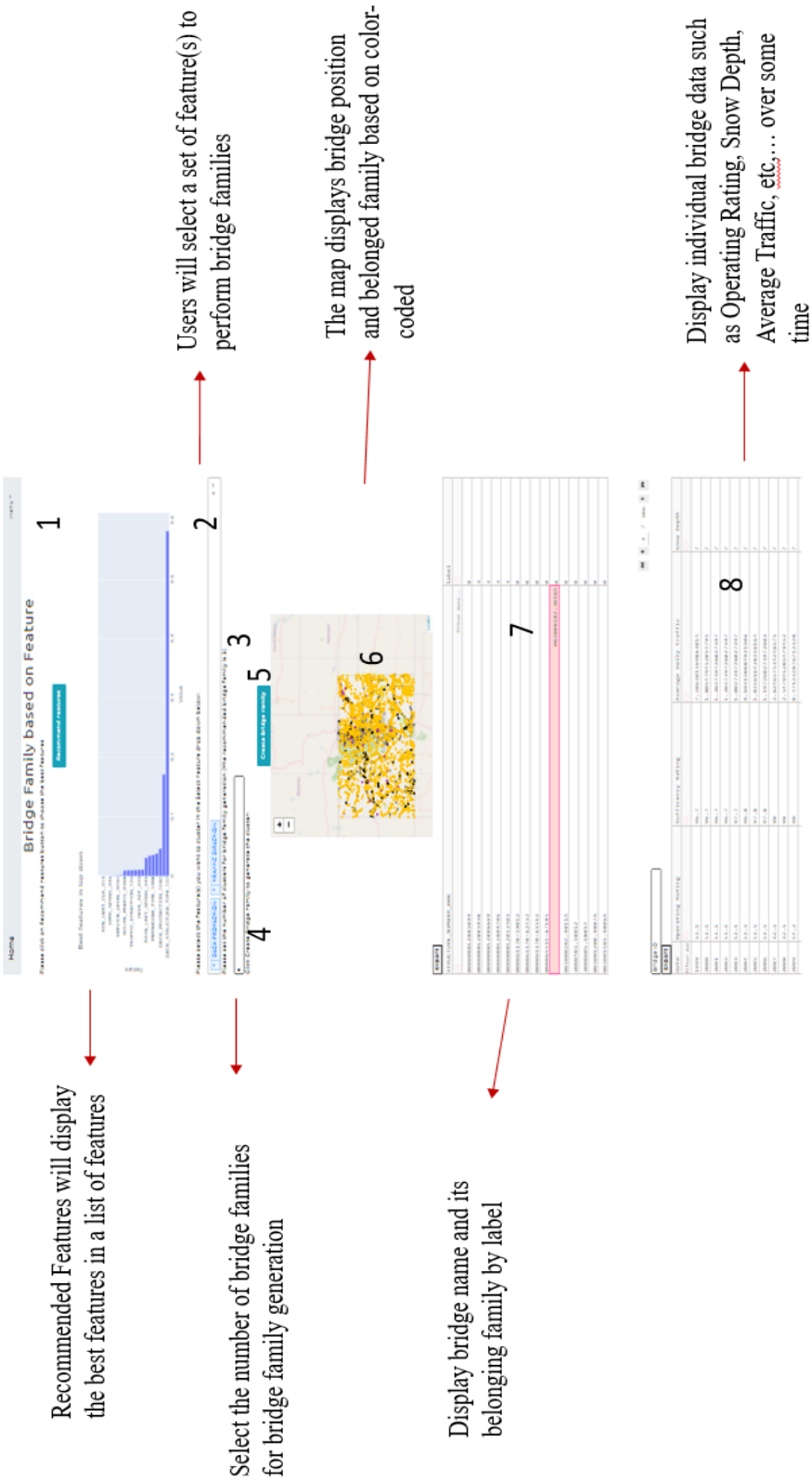

Figure 19: Procedure to generate bridge families using the developed tool

Figure 20 illustrates a sample result generated by the tool given the bridge dataset described in Section 2. As shown in the figure, in this case the entire dataset is clustered into 4 families showing a variety of performance patterns

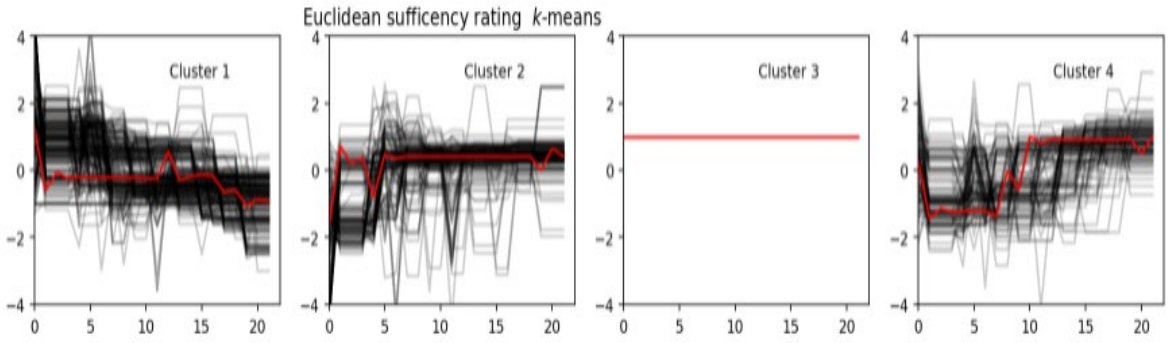

Figure 20: Sample result generated by the bridge family generation tool

5. Conclusions and Future Work

In this study, we proposed, developed, and evaluated a series of data-driven deep learning models for bridge deterioration forecasting and bridge family generation (or bridge subtyping) to be used by bridge engineers for effective bridge management. With extensive experimental evaluation using multi-modal real datasets including bridge performance data, traffic data and weather data for all bridges in Colorado, we have demonstrated that a selection of our proposed models significantly outperform existing models for the aforementioned two problems, respectively. Moreover, we have developed a standalone software package that allows bridge engineers to use these superior models for bridge deterioration forecasting and bridge family generation.

In the future, we plan to develop an end-to-end intelligent bridge management tool for bridge and culvert deterioration forecasting and anomaly detection to be used by CDOT bridge engineers as a tool for effective bridge management. This tool builds on and significantly extends our aforementioned work in this study in three ways: 1) integrating the deep learning models into a user-friendly software tool with graphical user interface and improved operational features to further enhance capabilities of the developed software package in this study, 2) developing enhanced physics-guided deep learning models that integrate traditional physics based bridge deterioration forecasting models with data-driven deep learning models for further improved performance in prediction of deterioration, and 3) introducing bridge performance anomaly detection as a new capability that allows for accurate prediction of bridge performance anomalies such as those that can lead to bridge failures/accidents.

6. References

[1] United States Department of Transportation Fed. Hw, "National Bridge Inventory," 1992. [Online]. Available: https://www.fhwa.dot.gov/bridge/nbi/ascii.cfm.

[2] National Center for Environmental Information, "Climate Data Online Search," 1981. [Online]. Available: https://www.ncdc.noaa.gov/cdo-web/datasets.

[3] I. Srikanth and M. Arocklasami, "Deterioration Models for Prediction of Remaining Useful Life of Timber and Concrete Bridges: A Review," *Journal of Traffic and Transportation Engineering,* 2019.

[4] H. Liu and Y. Zhang, "Bridge Condition Rating Data Modeling using Deep Learning Algorithm," *Structure and Infrastructure Engineering,* 2020.

[5] G. Morcous, "Comparing the Use of Artificial Neural Networks and Case-Based Reasoning in Modeling Bridge Deterioration," *Annual Conference of the Canadian Society for Civil Engineering,* 2002.

[6] O. Tokdemir, C. Ayvalik and J. Mohammadi, "Prediction of Highway Bridge Performance by Artificial Neural Networks and Genetic Algorithms," *2000 Proceedings of the 17th ISARC, Taipei, Taiwan,* 2000.

[7] E. Zivot, "Vector Autoregressive Models for Multivariate Time Series," *University of Washington Econ 584 Notes.*

[8] K. Gulden and G. Nese, "A Study on Multiple Linear Regression Analysis," *Procedia - Social and Behavioral Sciences,* vol. 106, pp. 234-240, 2013.

[9] D. Rumelart, G. Hinton and R. Williams, "Learning representations by back-propagating errors," *Nature,* vol. 323, pp. 533-536, 1986.

[10] I. Goodfellow and Y. Bengio, Deep Learning, MIT Press, 2016.

[11] C. Olah, "Understanding lstm networks".

[12] M. Schuster and K. Paliwal, "Bidirectional recurrent neural networks," *IEE Transactions on Signal Processing,* vol. 45, pp. 2673-2681, 1997.

[13] S. Bai, Z. Kolter and V. Koltun, "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling," *arXiv:1803.01271,* 2018.

[14] J. Long, E. Shelhamer and D. Trevor, "Fully convolutional networks for semantic segmentation," *CVPR,* 2015.

[15] F. Lassig, "Temporal Convolutional Networks and Forecasting," *Unit8,* 2021.

[16] Y. Zheng, Q. Liu, E. Chen, Y. Ge and J. Zhao, "Time Series Classification using Multi-Channels Deep Convolutional Neural Networks," *Lecture Notes in Computer Science,* p. 298*310.

[17] P. Chetti and H. Ali, "Analyzing the Structural Health of Civil Infrastructures using Correlation Networks and Population Analysis," in *The Eighth International Conference on Data Analytics*, 2019.

[18] K. &. C. S. Chang, "Bridge Clustering for Systematic Recognition of Damage Patterns on Bridge Elements," *Journal of Computing in Civil Engineering,* vol. 33, no. 5, 2019.

[19] P. Diez, "A clustering approach for structural health monitoring on bridges," *Journal of Civil Structural Health Monitoring,* vol. 6, no. 3, pp. 429-445, 2016.

[20] M. &. C. B. Knight, "Infrastructure Investigation Using Latent Class Cluster Analysis," in *International Conference on Computing in Civil Engineering*, 2005.

[21] E. Woods, "Tslearn, A Machine Learning Toolkit for Time Series Data," *Journal of Machine Learning Research,* vol. 21, pp. 1-6, 2020.

[22] S. M. S. D. F. a. H. K. Naveen Sai Madiraju, "Deep temporal clustering: Fully unsupervised learning of time-domain features," arXiv:1802.01059, 2018.

[23] W. M. S. D. a. H. B. Toon Van Craenendonck, "COBRASTS: A new approach to Semi-Supervised Clustering of Time Series," arXiv:1805.00779, 2018.

[24] Hearn, G., "Deteriorating Modeling for Highway Bridges", Structural Reliability in Bridge Engineering, Workshop a

[25] Rens, K. L., Nogueira, C. L., Transue, D. J., "Bridge Management and Nondestructive Evaluation", Journal of Performance of Constructed Facilities, Vol. 19, N. 1, 3-16, EUA, February, 2005.

[26] Rens, K. L., Nogueira, C. L., Neiman Y. M., Gruber, T., and Johnson, L. E., "Bridge Management System for the City and County of Denver", Practice Periodical on Structural Design and Construction, Vol. 4, N. 4, 131-136, EUA, November, 1999.

[27] Hartle, R. A., Amrhein, W. J., Wilson III, K. E., and Baughman, D. R., "Bridge Inspector's Training Manual 90", Federal Highway Administration, U. S. Department of Transportation, July 1991.

[28] REMR Technical Note OM-CI-1.2, "The REMR Condition Index, Condition Assessment for Maintenance Management of Civil Works Facilities", Suppl. 7, 1996.

[29] Stecker, J., Greimann, L. F., and Rens, K. L., "Chicago Harbor Lock Inspection", Technical Report submitted to U.S. Army Corps of Engineers, Chicago District, 1993.

[30] Greimann, L. F., Stecker, J., and Rens, K. L., "Inspection and Rating of Miter Lock Gates", Technical Report REMR-OM-7, 1990.

[31] Greimann, L. F., Stecker, J., and Rens, K. L., "REMR Management Systems Navigation Structures, Condition Rating Procedures for Sector Gates", REMR-OM-13, 1993.

[32] Greimann, L. F., Stecker, J., and Rens, K. L., "Training Manual for Inspection and Rating of Miter Gates, Sector Gates, Steel Sheet Pile, and Operating Equipment", U.S. Army Corps of Engineers, Construction Engineering Research Laboratories (CERL), Champagne, Illinois, 1992.

[33] Greimann, L. F., Stecker, J., Rens, K. L., and Nop, M., "REMR Management Systems Navigation Structures, User's Manual for Inspection and Rating Software, Version 2.0", REMR-OM-15, 1994.

[34] Hearn, G. and Shim, H.-S., "Integration of Nondestructive Evaluation Methods and Bridge Management Systems", Infrastr. Condition Assessment: Art, Science & Practice, ASCE Conference, Boston, 1997, pp. 464-473.

[35] Hearn, G. and Shim, H.-S., "Integration of Bridge Management Systems and Nondestructive Evaluations", Journal of Infrastructure Systems, Vol. 4, No. 2, June 1998, pp. 49-55.

[36] Bulusu, S. and Sinha, K. C., "Comparison of Methodologies to Predict Bridge Deterioration", Transportation Research Record 1597, 1997, pp. 34-42.

[37] White, III, C. C. and White, D. J., “Markov Decision Processes”, European Journal of Operational Research, 39, 1989, pp. 1-16.

[38] Scherer, W. T. and Glagola, D. M., “Markovian Model for Bridge Maintenance Management”, Journal of Transportation Research, Vol. 20, No. 1, 1994, pp. 37-51.

[39] Christof Angermueller, Tanel Pärnamaa, Leopold Parts, Oliver Stegle. Deep learning for computational biology. Molecular Systems Biology (2016) 12, 878.

[40] Benjamin Shickel, Patrick James Tighe, Parisa Rashidi. Deep EHR: A Survey of Recent Advances in Deep Learning Techniques for Electronic Health Record (EHR) Analysis. DOI:10.1109/JBHI.2017.2767063. IEEE Journal of Biomedical and Health Informatics, 2018.

[41] W. Byeon, T. Breuel, F. Raue, and M. Liwicki. Scene labeling with lstm recurrent neural networks. In CVPR, 2015.

[42] M. Chen and A. Hauptmann. Mosift: Recognizing human actions in surveillance videos. 2009.

[43] Y. Du, W. Wang, and L. Wang. Hierarchical recurrent neural network for skeleton based action recognition. In CVPR, 2015.

[44] R. Girshick. Fast r-cnn. In ICCV, 2015.

[45] N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using lstms. In ICML, 2015.

[46] S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, and K. Saenko. Translating videos to natural language using deep recurrent neural networks. arXiv:1412.4729, 2014

[47] Riccardo Miotto, Li Li, Joel T. Dudley. Deep Patient: An Unsupervised Representation to Predict the Future of Patients from the Electronic Health Records. DOI:10.1038/srep26094. Scientific reports, 2016.

[48] Nguyen, R. Alqurashi, Z. Raghebi, F. Banaei-kashani, A. C. Halbower, and T. Vu, “A Lightweight and Inexpensive In-ear Sensing System For Automatic Whole-night Sleep Stage Monitoring”, ACM Conference on Embedded Network Sensor Systems (SenSys 2016). New York, NY, 2016. Best Paper Award

[49] Nguyen, R. Alqurashi, Z. Raghebi, F. Banaei-kashani, A.C. Halbower, T. Dinh, and T. Vu, “In-ear Biosignal Recording System: A Wearable For Automatic Whole-night Sleep Staging”, Workshop on Wearable Systems and Applications (WearSys 2016).

[50] Farnoush B. Kashani, Gerard Medioni, Khanh Nguyen, Luciano Nocera, Cyrus Shahabi, Ruizhe Wang, Cesar E. Blanco, Yi-An Chen, Yu-Chen Chung, Beth Fisher, Sara Mulroy, Philip Requejo, Carolee Winstein. Monitoring mobility disorders at home using 3D visual sensors and mobile sensors. Proceedings of the 4th Conference on Wireless Health, Baltimore, Maryland, 2013.

[51 Max Lee, Farnoush Banaei-Kashani, “Addressing Overfitting in Convolutional Neural Networks for Classification of Multivariate Spatiotemporal Data”, Under review.

[52] Sridharan Raghavan, Farnoush Banaei-kashani, Seth Creasy, Ed Melanson, Leslie Lange, Michael A. Rosenberg, “Activity Classification for Patients with Ventricular Arrhythmia”, accepted for publication at PlosOne.