DOCTOR OF PHILOSOPHY

博士論文

Deep Sensing Approach to Media-Fusion Analysis for Modeling Bridge Dynamics

(橋梁の動態モデル構築のための深層計測による媒体統合解析)

Takaya Kawakatsu
川勝孝也

Department of Information and Communication Engineering,
Graduate School of Information Science and Technology,
THE UNIVERSITY OF TOKYO

東京大学情報理工学系研究科

December 9, 2018

Abstract

The most fundamental mission of modern computer systems is decision-making to assess the situation by consulting the sensing system and then command the actuator system to handle and control the situation. Such systems have been constructed for use in areas such as aviation, finance, and robotics by limiting their application to their own closed networks. Recently, with the widespread adoption of compact measuring instruments, the Internet, and small yet powerful processors, traditional computer systems are changing their concepts and roles. The Internet of Things and social cyber-physical systems (CPSs) are examples of such changes. We believe that society may become fully automated as a result of forms of artificial intelligences that handle heterogeneous sensor data collected from real-world observations. However, while decisions have been supported by experience and knowledge presented by many professionals, the wisdom thus obtained has not necessarily been recorded. To construct a CPS, we must develop a generic solution that systematizes the professional knowledge without depending on any languages or rules.

Frequently, deep artificial neural networks have attracted social attention by their superior performance and versatility with regard to various tasks of machine learning, but the essence of their superiority is that heuristics are excluded from the decision-making process. The applications of these networks are as yet limited to naïve tasks such as detection, classification, and regression, but they can already reduce the workload of professionals to a certain degree. For over three years, the research group at the National Institute of Informatics and I have searched for a way to apply neural networks to the problem of social infrastructure maintenance, and especially the task of assessing the safety of road bridges. The greatest difficulty in our research was the reversal of means and purpose. Initially, the target sensors should be installed under careful consideration of their use. Unfortunately, our study started with the task of finding some useful application of the sensor data that have been collected from real bridges over a long period. There was an initial strategy of realizing a fully automated system for monitoring the structural health of the bridges, but no tactics and concrete operations could be performed. The concept of deep sensing was born from such a barrier—one that is often faced in the area of data mining—and permitted us to move forward with our research.

Deep sensing is the effort of extracting rich latent information hidden in the observation data by aggressively applying neural network technologies to the sensor data. As we demonstrate in this dissertation, even trivial vibration data may be a rich source of information about passing vehicles. Our efforts may enlarge the role of the actual sensors installed on the bridge and may produce two major benefits. First, we may replace some specialized sensors by a versatile, miniature, and inexpensive sensor. Moreover, the necessity of developing specialized sensing devices may be reduced. Second, we may simplify entire sensing systems deployed in the real world by using the same single sensor for multiple purposes and by reducing the total number of sensing devices required. Deep sensing may be an antithesis of conventional sensor-fusion approaches that must synchronize many sensors accurately and are thus unreliable and require frequent inspection and repair. By applying the idea of deep sensing, we have successfully developed two approaches to detect anomalous events that may indicate some structural faults in, and damage to, the target bridge. One approach is the direct comparison of heterogeneous sensor data that extracts features from several data samples collected from heterogeneous sensors and media, including camera and strain sensors installed on the target bridge, and compares them in a common feature space shared by the data domains. The other approach is the modeling of bridge dynamics by a task of translation between sensors that extracts features from several data sources and estimates a signal that may be observed via another sensor. For these purposes, we have developed the techniques of spiral learning and of media-fusion generative adversarial networks (GANs).

Chapter 4 describes a video analysis framework for collecting ground-truth data for training the deep sensing models. The traffic surveillance system (TSS) that has been utilized for this purpose was proposed originally for the purpose of vehicle detection as a preprocessing step for analyzing natural vibration observed on bridges. Therefore, we also mention the analysis framework proposed for natural vibrations. The vibration response of a damaged bridge is known to have changed characteristics. To analyze the response, we must start by collecting waveforms of the vibration immediately following the passage of a vehicle. We then need to isolate just those vibrations caused by a single heavy vehicle if the vibration characteristics are to be accurate. Consequently, we developed the TSS that exploited a surveillance camera. The system identifies a vehicle from the video by combining a moving-object detector with an object detector based on a convolutional neural network (CNN), thereby estimating automatically the bridge's natural frequencies and damping ratios as features that characterize the bridge's damage.

Chapter 5 describes an application of deep sensing to the weighing task of vehicles passing a bridge. In this chapter, we introduce the bridge weigh-in-motion (BWIM) system. BWIM is a well-known technique for detecting overloaded vehicles crossing a bridge without requiring them to stop. BWIM may also be useful for monitoring the structural health of the bridge. To achieve accurate weighing of each vehicle, its properties such as speed, locus, and wheel positions, should be estimated in advance. Conventionally, such information has been obtained via additional sensors such as cameras or via peak-signal detection using multiple sensors installed across the bridge. This information may require substantial computational resources or expensive synchronization between sensors, and the complexity of the overall BWIM system may lead to frequent breakdowns. In Chapter 5, we propose a single-sensor-based BWIM system that utilizes a deep neural network. First, a vehicle's properties are obtained via feature extraction from the bridge strain response, as sampled by a single strain sensor. BWIM is then performed using the same response data. The model parameters for vehicle detection are optimized automatically by consulting a surveillance camera while obtaining ground-truth data for a large number of vehicles crossing the bridge. After the model is optimized for the target bridge, the camera may be removed. Our proposal paves the way toward low-cost, compact, single-sensor BWIM systems.

Chapter 6 proposes a damage detection framework for road bridges based on an anomaly detection technique. When a vehicle passes over a bridge, the bridge distorts in response to the vehicle's load. The response characteristics may change over time if the bridge suffers damage. We consider the detection of such anomalous responses using data from both traffic surveillance cameras and strain sensors. The camera data are utilized to treat each vehicle's identified properties as explanatory variables in the response model. The video and strain data are transformed into a common feature space to enable direct comparisons. This space is obtained via our proposed spiral learning method that is based on a deep convolutional neural network. We treat the squared Euclid distance between the video and strain data in the space as the anomaly score. We also propose an adversarial unsupervised learning technique for removing the influence of the weather from the video features. In our experiments, we found anomalous strain responses from a real bridge and were able to classify them into four major patterns. Unfortunately, we had no collection of ground-truth data for bridge damage detection; thus, we could not validate the meaning of the anomaly quantitatively. In this sense, our experiments in Chapter 6 are ambitious, and at present, the effectiveness of the proposal is unfortunately uncertain.

Chapter 7 describes an anomaly detection system that is more progressive than Chapter 6. The approach involves dynamic simulation whereby damage may be identified by detecting unusual mechanical behavior by the bridge components in response to passing vehicles. Conventionally, dynamic simulation requires expert knowledge of mechanics, materials, and structures, as well as accurate modeling. Moreover, dynamic simulation requires a detailed specification of the external forces applied, such as vehicle speeds, loci, and axle weights.

Chapter 7 introduces a novel media-fusion framework to obtain a dynamic model in a fully data-driven fashion. The proposed generative model successfully simulated strain responses for a real road bridge by consulting a camera and strain sensors on the bridge. The generative network was trained by an adversarial learning algorithm customized for media-fusion analysis. Moreover, anomalous sensor signals may be detected in terms of physical quantities rather than scalar anomaly scores introduced in Chapter 6. From the perspective of the interpretability of anomaly detection results, this approach may be superior to Chapter 6.

The three empirical studies are reviewed and discussed in Chapter 8, concluding this dissertation.

Although we mainly focus on the methodology of data mining, this dissertation contributes to informatics and to the research field of civil engineering. As a result, one of the papers establishing this dissertation was presented at an academic workshop on structural health monitoring. We believe our work will contribute to the wide acceptance of an autonomous maintenance system for social infrastructures such as bridges and buildings.

Acknowledgments

This dissertation was written with the support and efforts of many people. I would like to express my special thanks to everyone who gave me guidance, advice, and encouragement during my master and doctoral courses at the University of Tokyo and the National Institute of Informatics (NII).

First, I must thank my supervisors, Professor Dr Shinichi Sato and Professor Dr Jun Adachi. Professor Sato is a specialist who has led the research fields of recognition and information retrieval of video streams. He is also my official supervisor in my doctoral coursework, and his outstanding insight and friendly advice have encouraged my chosen research field, although my research topic was quite far from the mainstream of intelligent informatics. Professor Adachi has been my special supervisor throughout my graduate life at NII. He is a specialist in library and information science and has led Japanese academic societies of informatics for a long period of time. I am proud of being the last student at his laboratory. His inextinguishable curiosity and generous support allowed me to grow as a research specialist. Recently, I have felt lonely because I have fewer opportunities to discuss interests other than research with him.

I would like to thank my dissertation committee members, Professor Dr Kiyoharu Aizawa, Professor Dr Kenjiro Taura, and Associate Professor Dr Shunsuke Kamijo, for giving their time to provide constructive criticism and advice for my dissertation despite their busy schedules. Professor Taura was my supervisor during my undergraduate coursework at the Hongo campus and gave me much advice even after I graduated and left his laboratory. Professor Taura and Associate Professor Kamijo also served as my faculty advisors during my doctoral coursework.

I greatly appreciate the continued support of Professor Dr Atsuhiro Takasu during my years at NII. Through our frequent discussions in weekly meetings, he has supported my research directly as both my employer and substantial supervisor. Associate Professor Dr Kenro Aihara at NII also supported my research in its technical aspects including planning and maintenance of the computational resources, networks, and sensing systems. I am proud of the research collaboration with them.

The national research project *Infrastructure maintenance, renovation and management* from the cross-ministerial Strategic Innovation Promotion (SIP) program of the Cabinet Office, Government of Japan, supported the work in this dissertation. I have also received great assistance from many other people during this project. I particularly acknowledge the help of researchers and specialists including Professor Dr Saiji Fukada at Kanazawa University, Associate Professor Dr Takeshi Miyashita at the Nagaoka University of Technology, Associate Professor Dr Keigo Suzuki at the University of Fukui, and Mr Hiroshi Inada at Shimizu Corporation. Project Lecturer Dr Imura Hajime at the University of Tokyo has supported my research not only in my doctoral coursework, but also in my master coursework before he transferred from Hokkaido University. I also appreciate the assistance from Professor Dr Kazuo Imai and Ms Yuko Okamoto at NII, who have collaborated with me several times at exhibitions in the combined exhibition of advanced technologies (CEATEC) in 2016, 2017, and 2018.

I must express my most sincere gratitude to Dr Akira Kinoshita, who was a PhD candidate student at the Adachi laboratory and postdoctoral researcher at Takasu laboratory. I remember my manuscripts covered with his comments in red text following his peer review. This is probably a scene that all students (including me) who graduated from that laboratory will always remember vividly. Even after he left NII because of a job change, the manuscripts and materials he left behind have continued to teach me. I am proud of the days when I belonged to the laboratory as a doctoral student in the same period with him.

I would like to thank the secretaries at Adachi laboratory and Takasu Laboratory, Ms Ai Kuba, Ms Minako Nagao, Ms Ritsuko Hiraki, and Ms Akiko Takenaka. I would also like to thank my senior colleagues at Adachi laboratory, Mr Mitsuki Kimura, Mr Yimin Chu, Dr Kinoshita, Mr Naoki Kobayashi, and Mr Kazuki Murahashi; my peers Mr Nobuhito Haga and Mr Ryohei Nakashima, and my junior colleagues, Mr Kazushige Ando, Mr Toru Nishino, and Mr Akira Kakitani. Mr Haga was also my colleague at Taura Laboratory, and I heard much about his hobby of fishing during our discussions. I remember I recommended a camera (SONY RX100) as his fishing companion. He always brought his own lunch box to the laboratory, and I wanted to go out to restaurants to eat with him more often. Mr Nakashima was a contract researcher at Hitachi. I remember him, Dr Kinoshita, and I often going out to Jimbocho to eat, but I greatly regret that some of the restaurants there are now closed, such as Tokumanden, Imoya, and Tetsumen. Mr Ando was a famous person in the departments of electrical and electronic engineering, information and communication engineering (EEIC). I wanted to talk with him more on Twitter. Mr Nishino was an honest person and dedicated himself with enthusiasm to research on natural language processing. I remember his face when he was cheated on during a card game during summer camp at the Inose Rodge, NII at Karuizawa. Mr Kakitani is a highly cooperative person, but at the same time, he has a strong will. He continued to focus on theoretical research until the end of his master coursework, despite our advice. These individuals were all my valued colleagues, and I am pleased and proud to have spent my precious youth with them at the Adachi laboratory.

I am very grateful to all my friends at the Amateur Radio Club, University of Tokyo. They have encouraged and motived me to complete my PhD. Lastly, I appreciate the continued support, encouragement, and expectations of my family. I will fulfill my promises to my grandmothers and grandfathers by obtaining the honor of a PhD.

December 9, 2018

Takaya Kawakatsu

Contents

List of Figures

List of Tables

Chapter 1

Introduction

There is growing interest in the mining of large amounts of data acquired by huge sensing networks involving humans, consumer electronics, industrial machines, and society. Extremely large-scale data processing is realized by a backdrop of inexpensive, powerful, computational environments composed of parallel and distributed clusters. Modern computers can have large memories with hundreds of gigabytes per CPU socket. We believe that artificial intelligence with extensive knowledge will improve future human society. A cyber-physical system (CPS) is the realistic answer to this naive expectation.

1.1 Social Cyber-Physical System

The essence of the CPS concept is the application of a traditional information system to social decision-making. Historically, the fundamental mission of computational and communicational systems has been acceleration of decision-making. Fig. 1.1 shows the general architecture for decision-making, involving the sensor system and actuator system. In general terms, a decision system has four system components, namely data acquisition, situation analysis, command, and control. The first step, data acquisition, collects data from sensors through a communication network. The second step, situation analysis, supports the commander, by presenting possible operation plans. The third step, command, may contain operational planning for the long term, short term, and for real time. The fourth step, control, performs the command by controlling humans, machines, and actuators. Such a decision system has been applied to every field of human activity, including finance, industry, logistics, aviation, diplomacy, and security. Most decision systems have developed individually within their specific field of application. For example, an avionics system cannot control a ship or buy and sell stocks. By limiting their situations, missions, and responsibility levels, decision systems have been put into practical use.

A social decision system may integrate specialized information systems by organizing and sharing resources for sensing, communication, computation, situation analysis, and actuation, which are deployed and interconnected widely in the real world and work together organically via the Internet. As a result, the social decision system is responsible for any problems and applications in society. This responsibility may be extended to economic and political fields in the distant future. In this dissertation, we are interested in the problem of the maintenance of social infrastructures. It is important, from the perspective of improving social welfare and security, to maintain regions where depopulation has occurred as a result of people moving to urban areas. In this context, the social decision system may produce great benefits. We expect that the system may resolve the problem of some social requirements that have been neglected for years because of their expense being not worth their benefits, e.g., maintenance of old regional roads used by few people.

Unfortunately, the CPS framework is new and technically ambitious. Although we labeled a CPS as a decision-making system, the main research interest is situation analysis, which is just a portion of the four major components of a CPS. This dissertation is not an exception. Strictly, the main topic of this dissertation is developing the techniques and frameworks for bridge damage detection, which will be a trivial component implemented on the CPS framework in the distant future.

1.2 Structural Health Monitoring

In this dissertation, we focus mainly on the problem of structural health monitoring (SHM), especially the case of road bridges. Many bridges built in Japan in the 1960s have deteriorated and require substantial inspection. As reported by the Japanese government [76], there are at least 700,000 road bridges that are more than two meters long, and 75% of them are managed by municipal offices. In addition, in 2013, 18% of the 700,000 bridges were constructed over 50 years previously and this percentage could rise to 43% in 2023 [76]. Unfortunately, periodical bridge inspection is costly, and hiring civil engineers all over the nation (or world) may be difficult. Therefore, bridge owners and maintainers demand health-monitoring systems that can automate the inspection process completely or partially.

Information scientists can approach the health-monitoring problem by applying the concept of social CPSs. First, we install a number of inexpensive, miniature sensors on bridges. Second, we acquire sensor data of bridge dynamics via the Internet and manage them in a multimedia database system. Third, we apply some techniques of data mining and machine learning to assess the risk of bridge collapse. Finally, the bridge owner makes decisions regarding bridge maintenance, including repair, demolition, and renovation.

By using long-term monitoring data, we aim to detect small signs of internal damage to the bridge, such as the fracture and corrosion of steel cables, frames, and reinforcing bars, which may cause bridge collapse. Such damage may be hardly detectable by visual and hammering inspection. In contrast, we utilize accelerometers, transducers, and cameras, considering their versatility and expense. Compared to direct sensors such as radiation inspection, these sensors have a great weakness in that they cannot identify internal damage. However, these sensors have great merits in that they are inexpensive, durable, weatherproof, and suitable for permanent measurement. These benefits may make our monitoring system superior to other approaches, including running inspection vehicles [51], in terms of the amount of acquired data used for assessing damage.

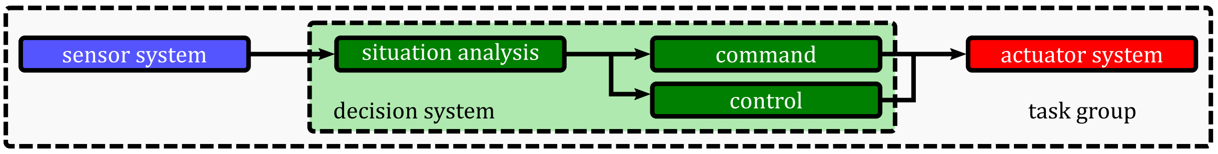


Figure 1.1: General mechanism for decision-making.

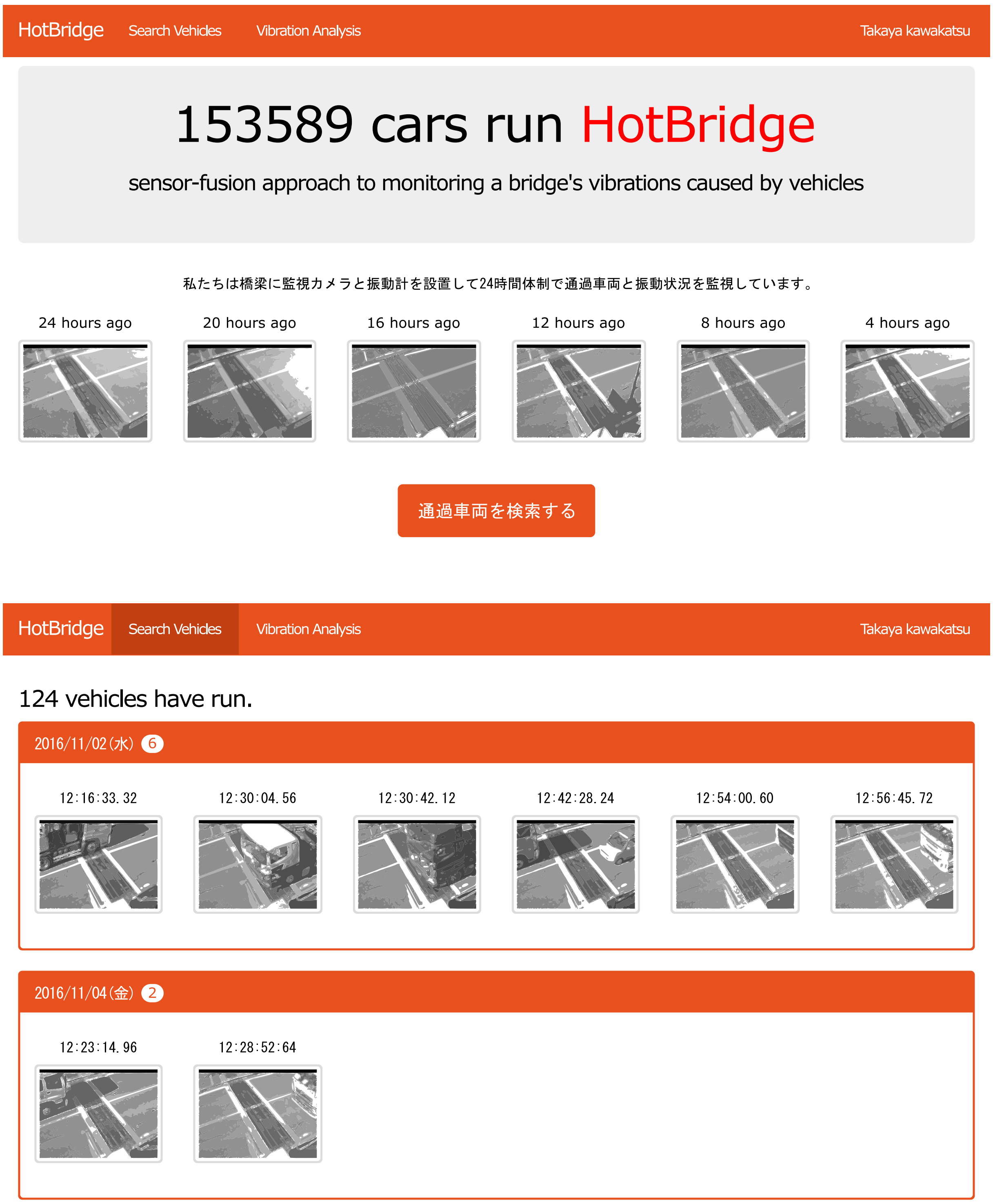
1.3 Integrated Health Monitoring System

The main goal of this research is to provide an integrated data management system for the SHM of social infrastructures. The targets are not limited to bridges but include tunnels, roads, and other types of buildings. This system collects and archives data from heterogeneous data sources in an integrated database management system (DBMS). The managed data include sensor data, videos, inspection documents, and geological and meteorological data. They are managed in a group of database (DB) systems, including relational, spatiotemporal, and file DBs.

We are aiming at a highly scalable SHM system based on a CPS where various talents from various fields, e.g., owners, maintainers, civil engineers, and data scientists, cooperate with each other. For maintainers and civil engineers, the system provides a web interface for data search, visualization, and annotation. For data scientists, the system provides an interactive analysis interface including programming environments. For owners, the system provides an alert function implemented by the analysts. The processes of exploratory analysis are shared as documents that are managed in the integrated DBMS and utilized as toolkits for data mining, visualization, and alerts.

As a part of the SIP [80] program, the research members of NII, including the author, have implemented a prototype named integrated data management platform (IDAMP). The IDAMP is composed of a backend and a frontend. The backend integrates heterogeneous DBMSs including MongoDB [57] and MariaDB [52] and provides query application programming interfaces (APIs). Inside the backend, a traffic surveillance system (TSS) proposed in Chapter 4 is utilized for indexing vehicle passing events.

IDAMP has two types of frontends, namely a programming interface and a web interface. The programming interface is based on Jupyter [69] and provides Python APIs for data scientists. The web interface provides a search engine for civil engineers. Fig. 1.2 illustrates the web interface named joint data system (JDS) HotBridge, which was developed by the author. Users can search vehicles that passed over the target bridge by specifying the time ranges, number of axles, and lanes. Then, users can check the sensor signals and video by selecting listed thumbnails. In future work, the JDS may provide extensions implemented inside the



153589 cars run HotBridge

sensor-fusion approach to monitoring a bridge's vibrations caused by vehicles

私たちは橋梁に監視カメラと振動計を設置して24時間体制で通過車両と振動状況を監視しています。

124 vehicles have run.

2016/11/02(水) 6

12:16:33.32 12:30:04.56 12:30:42.12 12:42:28.24 12:54:00.60 12:56:45.72

2016/11/04(金) 2

12:23:14.96 12:28:52:64

Figure 1.2: The frontend web interface JDS HotBridge for bridge health monitoring.

programming interfaces. We expect the contributions of this dissertation will be exploited widely by civil engineers and maintainers through IDAMP.

1.4 Research Questions and Objectives

The fundamental interest of this study is in applying deep learning techniques to the problem of SHM for the purpose of creating social CPSs, which acquire bridge dynamical behaviors sampled by heterogeneous sensors deployed on a real bridge. A neural network may achieve flexible application to any bridges simply by the acquisition of training data and automatic optimization based on a statistical loss function. By taking this advantage, we may construct a fully data-driven health-monitoring system backed by a large amount of observation data, and may exclude expert heuristics from the process of bridge assessment. This study's challenge is composed of two steps. First, we determine the latent features involved in a single sensor that may explain much about external forces applied to the bridge components. Second, we develop dynamical models for target bridges that may predict the bridge's dynamical behavior by consulting sensors deployed on the bridges. Unfortunately, the mechanisms for damage progression on real bridges are yet to be fully clarified. There has been no collection of large-scale ground-truth data for damage identification, which would be helpful for approaches involving data mining. Therefore, we must develop completely new criteria for damage detection based on anomaly detection approaches [33, 31].

1.5 Analysis Tasks and Methodology

In this dissertation, the following four analysis tasks are examined: (1) video analysis of a surveillance camera for traffic dataset preparation, (2) vehicle detection using a single strain or acceleration sensor on bridges, (3) anomaly detection of strain responses caused by vehicles, (4) strain response prediction using a generative model.

The first task was traffic dataset preparation, which was utilized for three other tasks as training data. The task was conducted on real video data recorded by a traffic surveillance camera installed at the bridge entrance. The camera had captured millions of vehicles for years, and we implemented a traffic detection system based on a deep CNN. The traffic surveillance system (TSS) was first proposed for natural vibration analysis, backed by the expert knowledge that a damaged component may change its natural frequencies. Consequently, TSS was diverted to another purpose: the preparation of datasets for training and evaluation of the following three tasks.

The second task was vehicle detection from strain and acceleration signals obtained as dynamic responses to vehicles crossing bridges. The task was conducted on real strain and vibration data recorded by strain meters and an acceleration sensor installed underneath bridge decks. The proposed system was trained using the traffic dataset created by TSS as ground truth. The proposal successfully detected vehicles running in specific lanes and estimated their properties including their speed, locus, and number of wheel axles. This rich signal data was collected by each sensor instead of consulting multiple sensors installed at multiple points. Our system worked as if it had consulted a virtual sensor that could detect vehicles directly. We named this approach *deep sensing*. The obtained model may be utilized for a vehicle-weighing system, called bridge weigh-in-motion (BWIM) [47, 99], which has been utilized for bridge health assessment and legal enforcement of road traffic regulations. Moreover, the proposed CNN was utilized for the following two tasks.

The third task was the anomaly detection of strain responses caused by vehicles. The task was conducted using both the video data and strain sensor data in a media-fusion fashion. The task exploits the achievement of the second task that succeeded in estimating vehicle specifications from bridge strain responses. The task assumes a response function which explains the bridge response by consulting vehicle properties and a common feature space shared by two domains of vehicle appearances and strain responses. We developed the *spiral learning* technique for obtaining the common feature space to compare the video and sensor data directly. We defined the estrangement of video and sensor data as *anomaly score* and found some anomalous strain responses and classified them into several cases.

The fourth task was strain signal prediction using a generative neural network. The task was conducted using both the video and strain data in a media-fusion fashion. A multimodal generator was developed, which takes a surveillance video and a signal sequence obtained by a single strain sensor as input and estimates another signal sequence that should be observed at another point. The generative model may predict strain responses caused by specific vehicles by consulting a camera and strain sensor. The video data was exploited to improve the estimation quality by providing rich information about target vehicles, including shapes, axle positions, speeds, and loci. The sensor data was exploited to obtain data on axle loads, which was an aspect that could not be detected using video data. The optimized generator may be utilized for detecting anomalous responses by comparing the predicted and observed waveforms for each vehicle.

1.6 Structure of the Dissertation

This dissertation is organized as follows.

Chapter 2 introduces sensor data analysis tasks and methods and related work.
Chapter 3 introduces data analysis techniques for image and video, including CNNs, GANs, and object detection and tracking techniques, which can be applied to the task of vehicle detection using a surveillance camera.

Chapter 4 presents the framework of traffic analysis based on image processing techniques. The proposed system was originally developed for natural vibration analysis, and this chapter also presents the vibration analysis framework. This chapter includes the contents of the article [32].

Chapter 5 presents the deep sensing approach. Three CNN models were designed for vehicle detection, and we successfully exploited the information richness of each sensor with high accuracy. The proposed approach was evaluated for two real bridges in Japan. The possible application of the proposed system is BWIM, and our approach may realize an inexpensive, durable BWIM system using only a single sensor. This chapter includes the contents of the articles [34, 35].

Chapter 6 presents the spiral learning approach to directly compare traffic surveillance video and strain response data. A multimedia CNN was designed by merging two CNNs for video and sensor data and was optimized via the spiral learning proposal. We present some case studies on anomalous vehicles identified by the anomaly scores proposed in this chapter. This chapter includes the contents of the article [33].

Chapter 7 presents the media-fusion mechanism of bridge dynamic modeling. A generative CNN was designed by improving the multimedia CNN proposed in Chapter 6 for the purpose of sensor-to-sensor translation. This chapter includes the contents of the article [31].

Chapter 8 reviews and discusses the five studies [32, 34, 35, 33, 31], and this dissertation concludes with future perspectives.

Chapter 2
Structural Health Monitoring

Many road bridges built in Japan in the 1960s have deteriorated and now require substantial inspection. This chapter reviews SHM using sensors installed on target bridges. The introduced techniques aim to identify small signs of bridge deterioration by developing bridge dynamic models, driven by models and by data.

2.1 Overview

Unfortunately, the true mechanisms for damage progression on real bridges are yet to be fully clarified. Therefore, we have addressed the bridge problem not by preventing damage progression but by identifying damage.

There are two major approaches to bridge damage detection, namely destructive inspection and nondestructive inspection. Inspection techniques that do not involve disassembly of bridge components, such as peeling off the concrete surfaces, are called nondestructive inspection techniques. To realize inexpensive, frequent inspections for bridges in service, nondestructive approaches must be developed.

Traditionally, there are four major approaches to nondestructive damage detection, namely visual inspection, hammering inspection, radiographic inspection, and ultrasonic inspection. Among the four approaches, the former two techniques can be performed without any expensive, large-scale equipment. However, they are mainly targeted to bridge surfaces and can hardly identify internal damage inside the bridge components. To investigate inside the bridge, we must remove surface components such as concrete. The techniques of radiographic and ultrasonic inspection have been utilized for nondestructive identification of fractures and the corrosion of reinforcing steel rods, girders, and decks. These techniques require dedicated equipment including transmitters, receivers, and image processors, as well as bridge experts.

With the widespread adoption of compact sensing devices, attention is being directed toward a fully automated approach using sensor fusion involving heterogeneous sensors installed on the target bridge. One well-known example is crack detection on bridge surfaces via camera and image recognition based on CNN [106], but this method is mainly targeted at damage to bridge surfaces. The definitive approach to internal damage detection is bridge dynamic inspection, which investigates the mechanical characteristics of bridge components. Such a technique analyzes dynamic responses to external forces affecting the bridge, such as wind, earthquakes, and passing vehicles. In this context, indirect bridge monitoring [61, 51] has been developed, which uses an instrumented vehicle and collects the dynamic properties of bridge structures from the dynamic response to the instrumented vehicle while crossing the target bridge.

2.2 Bridge Dynamic Analysis

Studies of damage detection based on dynamic response can be classified according to two major approaches. The first approach uses long-term behavior of the bridge, such as natural vibration [2, 8], which does not depend strongly on individual passing vehicles. The second approach uses transient state analysis, which identifies an unusual dynamic movement by the bridge components in response to every passing vehicle [10, 27]. Compared with the first approach, the second approach may provide rich information sampled under a variety of traffic conditions. To detect suspicious mechanical responses, the dynamic system of the target bridge must be modeled in advance.

Existing techniques for bridge modeling can be classified according to two major approaches, namely explicit mechanical modeling and implicit mechanical modeling. The explicit modeling approaches utilize finite element analysis (FEA) [95, 82, 56, 55]. They are typically created by hand and optimized by test-vehicle runs and iterative model updates [95], typically using sensors installed on the bridge. It is usual for a bridge to bend and distort as a vehicle crosses it. If we assume that all vehicle properties including speed, locus, axle positions and loads, could be collected in advance, the structural response of the bridge, including strain, displacement, and vibration, would be predictable by using a bridge mechanical model. By comparing the undamaged and damaged FEA models [82], the damage can be localized. However, FEA requires accurate model making, which is not feasible for the majority of existing bridges.

In contrast, the implicit modeling approach abandons explicit model construction, because accurate FEA modeling is costly. There are two main methods for anomaly detection. The first method is based on using the model parameters that dominate the bridge's dynamics, including natural vibration frequencies [2], damping ratios [8, 32], and influence lines [10, 27]. The second method is based on physical quantities where the anomaly is defined as a dissociation between observed sensor data and predictions. The predicted data can be quasistatic [45, 48] or dynamic [105]. Traditionally, the transient signals are explained by using Kalman filtering [3, 96, 70, 64] for the case of quasistatic linear signals. In a recent approach, Neves et al. [58, 59] modeled bridge acceleration signals by using a perceptron that took previous 5-gram acceleration samples, axle loads, and axle positions as its input. Compared to FEA, the anomaly detection approach has a great disadvantage in cases where the damage has occurred before model construction and therefore cannot be completed.

2.3 Natural Vibration Analysis

Damage detection from natural vibration at bridge components is a well-known, classical approach. Recently, Li et al. [43] evaluated the effectiveness of a natural-frequency-based damage detection approach, while Koo et al. [41] assumed a simple damped oscillation model and found that the damping ratio degrades with damage. Historically, Bicanic and Chen. [2] considered a bridge's characteristic equation, as shown in Eq. (2.1).

$$(K - \lambda_i M)\phi_i = 0, \tag{2.1}$$

where K and M are the stiffness matrix and mass matrix, respectively, and λ_i and ϕ_i are the i-th eigenvalue and mode shape for the bridge's structure, respectively. The bridge's damage is regarded as a difference ΔK in the matrix K, and the Eq. (2.1) is to be translated into Eq. (2.2).

$$\{K + \Delta K - (\lambda_i + \Delta\lambda_i)M\}(\phi_i + \Delta\phi_i) = 0. \tag{2.2}$$

The change in K is to be observed in the change of the bridge's natural frequency. In fact, the natural frequency varies to a certain degree, and Cornwell et al. [11] evaluated the relationship between frequency and temperature. Therefore, Magalhaes et al. [50] used a linear regression model to cancel the effect of the temperature–frequency relationship, while Jin et al. [29] used a multilayer perceptron.

In our research group, Kakitani has improved the natural-frequency-based approach by considering the bridge's vibration response to be a mixture of damped oscillations, as described in Eq. (2.3). He mainly focused on the bridge's natural vibration and applied his mixture model to the actual vibration waveform observed on an expressway by utilizing the matrix pencil method [26, 79] as a means of regression analysis. His model handles not only natural frequencies, but also time constants corresponding to the natural frequencies as features that represent the bridge's damage.

$$y(t) = \sum_{k=1}^{K} R_k \exp\left\{\left(-\frac{1}{\tau_k} + i2\pi f_k\right)t\right\} + n(t), \tag{2.3}$$

where R_k, τ_k, and f_k represent amplitude, time constant, and frequency of the k-th exponential function, respectively. $n(t)$ is the noise.

2.4 Bridge Weighing in Motion

Vehicle weighing has two aims: to enforce laws regarding illegally overloaded vehicles [74] and to estimate the damage progression of bridges [7]. Although the mechanisms for damage

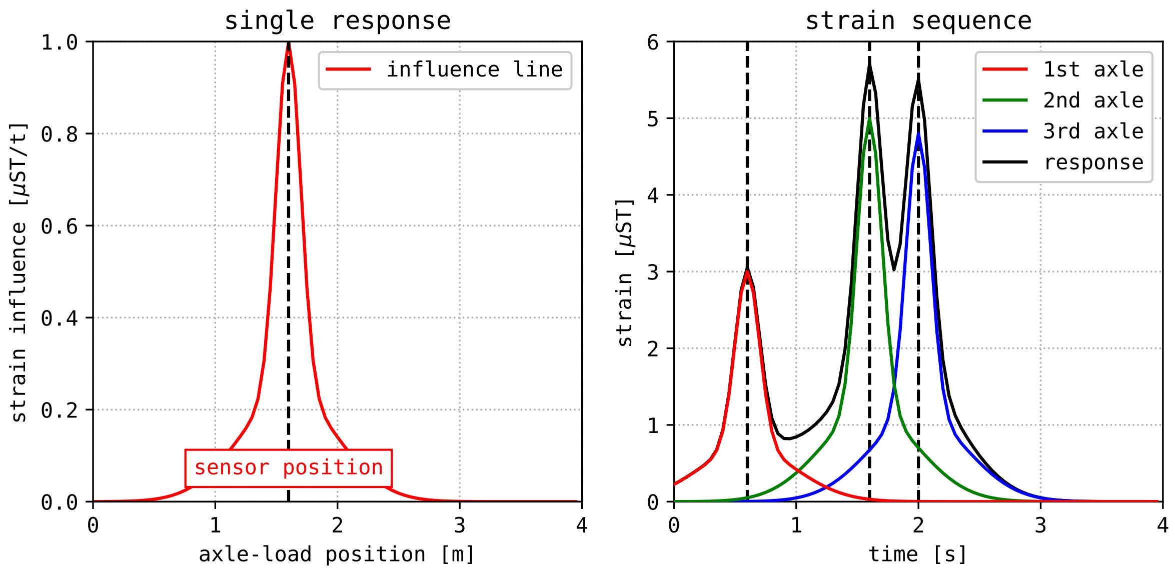


Figure 2.1: Mixture of influence lines scaled by axle weights.

progression in real bridges are yet to be fully explained, an accumulated volume of heavy vehicles appears to correlate with structural damage. To demonstrate the hypothesis, many researchers have addressed the task of detecting heavy vehicles. One simple approach is to install an axle load meter on the road surface. However, this meter is difficult to retrofit to existing bridges because this process requires paving work for installation. Moreover, this meter is fragile and requires frequent repair. Additionally, accurate weighing may impose limits on traveling speed. BWIM systems [42, 39] address this problem by treating the bridge structure itself as a large weighing scale. BWIM utilizes data from a strain sensor installed at the bridge deck, girder, beam, or flange.

Once a vehicle enters the bridge, the strain sensors detect the deflection of the bridge's beam or deck and estimate the weight of the vehicle, as shown in Fig. 2.1. BWIM can then estimate the axle loads in consultation with additional information about the vehicle, including its speed, locus, and axle positions. BWIM assumes a linear response model [47, 99] where the strain measurement $s(t)$ at time t is proportional to the product of axle weight $w(x,t)$ and the value of the influence line $i(x)$ at axle point x:

$$s(t) \approx \hat{s}(t) = \int_0^l w(x,t)i(x)dx, \tag{2.4}$$

where l is the bridge length. The function $i(x)$ indicates the proportionality factor for $w(x,t)$ and is measured by tests run in advance. The objective $w(x,t)$ is obtained by minimizing the square of the difference between ground truth $s(t)$ and the predicted $\hat{s}(t)$.

To obtain an accurate result, we require information about the vehicle's trajectory and shape at the time. Typically, vehicle lane, speed, and axle positions are required.

Locus information is sometimes desirable, such as when the axle position across the lane (orthogonal to the bridge axis) has a great influence on the strain measurements. Conventionally, vehicle speed and axle positions have been obtained by additional sensors at the road surface, such as cameras [63]. The speed can be estimated from the time lag of the strain (or vibration) peaks sampled at two positions on the bridge. The axle positions can then be estimated from the peak times by assuming uniform linear motion at the determined speed. That assumption may be a severe limitation for real applications because a vehicle may accelerate or decelerate. Moreover, such a system is hardly applicable to small bridges and requires strict synchronization between the sensors, tending to make the system costly, fragile, and unreliable.

2.5 Vehicle Detection Sensors

Recently, the concept of a nothing-on-road (NOR) BWIM [81, 107, 30, 23, 100] has been proposed, and this concept uses sensors beneath the road surface for vehicle detection. This strategy may prove to be a definitive vehicle detection technique based on the sensors' durability, weatherability, and ease of installation on existing bridges. Potentially, the NOR approach may simplify the total system radically because it may be possible to merge the main strain sensor for weighing with the additional sensors for vehicle detection. Unfortunately, existing NOR systems can hardly reduce the number of sensors because they employ a naïve peak-to-peak approach for speed estimation that consults multiple sensors. For example, Sekiya et al. [81] utilized two accelerometers installed at the bridge entrance and exit. Zhao et al. [107] proposed a NOR system that counts pulses sampled by multiple transducers beneath the deck slab. However, these methods are heuristic, with the thresholds for ignoring noise impulses having to be optimized by hand. Kalhori et al. [30] have also used shear-strain measurements for axle detection because the global flexural strain is less sensitive to individual axles. He et al. [23] proposed a novel technique to overcome this limitation by assuming a virtual simply supported beam. Interestingly, Yu et al. [100] demonstrated single-sensor-based speed estimation and axle detection by using wavelet analysis. They demonstrated that while the global bending response conceals information about each wheel axle's appearance and disappearance on the bridge span, the information could be revealed by multiresolution wavelet transforms. Although their approach might be an effective option in contrast to our proposal, the wavelet resolution must be optimized heuristically, and their approach cannot estimate the traveling lane and locus by itself.

Some BWIM systems (e.g., [81]) use an accelerometer instead of a strain sensor because an accelerometer is less expensive and longer-lived than a typical strain sensor. Such a system calculates bridge displacement by integrating the acceleration response twice. Because an accelerometer may capture small vibrations caused by distant vehicles and contains a constant offset component, the accelerometer-based approach is much more complex, heuristic, and impractical at the present time. However, because we believe this approach may become an alternative to strain-based BWIM systems at some stage, we consider the problem of vehicle detection in terms of strain sensing and using accelerometers (see Section 5.3).

The most controversial issue in previous BWIM research is that many studies lack any evaluation in practical use under heavy traffic conditions. Their accuracy is demonstrated only by simulation [42] or test vehicle experiments [81]. Because of the number of aging bridges in the world (especially in Japan), the lack of real investment is obvious. To overcome this, we need an evaluation framework for BWIM systems that automatically collect massive amounts of traffic data.

2.6 Influence Line Estimation

Aspects of bridge dynamics such as bending, strain, and displacement can be estimated by introducing the concept of an influence line. As described in Section 2.4, BWIM [47, 99] utilizes the linear response model for vehicle weighing where the strain measurement $s(t)$ at time t is proportional to the product of axle weight and the value of the influence line. Such a linear response may be observed at various bridge components, including flanges, beams, and decks.

In this section, we define an influence line $i(t)$ as a function of time t instead of the original form $i(x)$ at axle position x. We assume that the influence line $i(t)$ was obtained by a test run using a vehicle with K axles traveling at a fixed speed v. If k-th axle was a weight of w_k and passed over the strain sensor at time τ_k, the estimated strain $\hat{s}(t)$ may be defined as Eq. (2.5).

$$\hat{s}(t) = \sum_{k=1}^{K} w_k i(t - \tau_k). \tag{2.5}$$

Tateishi et al. [89] proposed an algorithm to obtain the influence line $i(t)$ by using a Fourier transform. First, Eq. (2.5) can be transformed to a function $S(\omega)$ of frequency ω in the Fourier domain as Eq. (2.6).

$$S(\omega) = I(\omega) \sum_{k=1}^{K} w_k e^{-j\omega\tau_k}, \tag{2.6}$$

where j is the imaginary unit, e is the Napier's constant, and $I(\omega)$ is the Fourier transform of the influence line $i(t)$. By deformation of the formula, the influence line $i(t)$ in the time domain can be obtained as Eq. (2.7).

$$i(t) = \mathcal{F}^{-1} \left[\left(\sum_{k=1}^{K} w_k e^{-j\omega\tau_k} \right)^{-1} S(\omega) \right]. \tag{2.7}$$

In Section 5.5, we utilize this algorithm for an estimation of 32 influence lines.

Some researchers [10, 27] have applied the linear response model to bridge damage detection. Damage may be detectable by detecting anomalous strain responses or extracting a temporal influence line from sensor data. This approach needs vehicle properties including speed, loci, axle positions, and weights for every passing vehicle.

Zaurin and Catbas [101, 102] investigated the collection of vehicle properties via surveillance camera analysis. A problem with their approach is that the target vehicles are limited to test vehicles with known axle weights. The most obvious approach to axle weighing is to use an axle load meter. However, as discussed in Section 2.4, this approach is difficult to retrofit to existing bridges, requires frequent repair, and may impose limits on traveling speed on the bridges. An alternative solution uses a BWIM [89, 65] system. However, if the bridge becomes damaged, the influence line may change and lead to unreliable axle load estimates. The influence line may also change its shape if the running position in the lane changes. We must therefore develop a complex model to handle the large number and wide variety of patterns of strain responses collected by test runs in advance.

Chapter 3
Video Analysis Technology

This dissertation exploits several techniques for video processing, including object detection and feature extraction for experiments in later chapters. In Chapter 4, we propose a traffic surveillance system (TSS) for an analysis task of traffic on bridges. In Chapter 6, we propose an anomaly detection technique that compares vehicle appearance and strain response on the bridge. In Chapter 7, we propose a strain prediction technique that considers vehicle appearance as an explanatory variable. In this chapter, we explore the basic technologies such as CNNs, GANs, and their applications such as object detection and tracking.

3.1 Convolutional Neural Networks

A neural network is a computation model that consists of multiple layers of neurons. Each neuron has parameters such as weight $\boldsymbol{w}$ and bias b that are optimized through back-propagation learning [78] and applies an activation function f to its input z:

$$\boldsymbol{y} = f(z) = f({}^{t}\boldsymbol{w}\boldsymbol{x} + b), \tag{3.1}$$

where $\boldsymbol{y}$ is the output value and z is an intermediate value. Each layer plays a role in some nonlinear mapping, and a multilayer perceptron, one of the simplest networks, can approximate any continuous function [17]. Generally, a deeper network can extract more complex features from data; however, the extraction may increase the total number of weight parameters and make the training more difficult. Many systems [84, 21] have achieved remarkable results on image classification problems by using CNNs, which can decrease the number of network parameters drastically by using convolution layers that behave in a similar way to digital filters. Fig. 3.1 illustrates an example mechanism of a CNN, which is composed of a convolution, padding, and max pooling. Unfortunately, the network depth (number of the layers) has been restricted because of attenuating gradients for the loss function in each layer during back propagation. This attenuation of gradients is called a vanishing gradient. As a countermeasure, He et al. [21] proposed using a convolution layer block called a *residual block* to solve this problem. A residual block divides the propagation path into two branches, with one feeding two tandem plain convolutional layers and the second being a shortcut to the block output. A CNN that has a residual block, called a ResNet, works as if two shallower and deeper CNNs share several layers in common, helping prevent underfitting. A practical ResNet involves many convolution layers and residual blocks, e.g., two convolution layers and 50 residual blocks [21].

One of the strengths of neural networks is that they require less prior knowledge of data distribution than other machine-learning models, such as Gaussian mixture models or support vector machines. Instead, we must define several hyperparameters, e.g., the network depth, width, activation functions, and the optimizer. Therefore, many researchers have proposed various architectures in accordance with the target data.

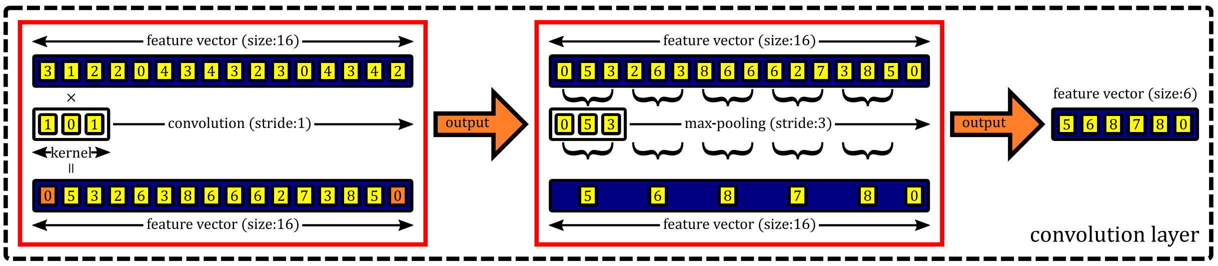


Figure 3.1: Convolutional layer mechanism.

3.2 Generative Adversarial Network

Recently, generative models [38, 20] are attracting much interest from many researchers. They have been introduced for image generation, which takes a random variable as input and generates a fake image that resembles a real one. Variational auto encoder networks (VAEs) [38] and generative adversarial networks (GANs) [20] are the representative models of generative neural networks.

A VAE network is descended from the auto encoder [25], which was originally introduced for the dimensionality reduction of vector data. The concept of auto encoding is quite simple. A neural network called an *encoder* takes raw vector data and outputs an encoded feature vector which may contain some latent variables explaining the input data. Then, another neural network called a *decoder* accepts the encoded feature and reconstructs the original vector data which are fed to the encoder. Both encoder and decoder networks are trained to minimize the difference between the original and regenerated data for the training data. A VAE network [38] exploits the decoder network as a generator model. It assumes some simple distribution for the latent variables, e.g., Gaussian mixture models, and the encoder network outputs the distribution parameters, e.g., mean and variance, for each sample. Then, a random variable is generated for each sample by using the distribution parameters, and the decoder network reconstructs the original data. Both encoder and decoder networks are trained so that the variational lower boundary for the log likelihood of the original data is maximized.

A GAN [20] comprises two neural networks, namely a generator network and adversary discriminator network. The generator creates images that are exactly like the real ones, and the discriminator may then misjudge the created images as real images. The discriminator finds fake images from a given image set that includes real and generated images. Compared to a VAE, a GAN does not learn any explicit distribution parameters for the latent random variable. Instead, the generator is trained so it deceives the adversary model well. The mechanism is as follows. First, the generator accepts a random variable, typically sampled from a Gaussian distribution, and generates fake data. Next, real observations and fake samples created by the generator network are merged into a dataset. The adversary discriminator is trained so it successfully discriminates the fake samples from real observations. Finally, the generator network is trained so that it generates fake data of high quality that is hardly distinguishable as fake for the adversary network. The training mechanism can be described as a discriminator aiming to minimize the discrimination error, whereas the generator aims to maximize it. The training process is repeated until the discriminator fails to find any faults in the fake data. Interestingly, a GAN may produce data for specific classes by introducing conditional factors [62].

3.3 Techniques for Vehicle Detection

Background subtraction is an old approach to extracting the mask image of a moving object from a video. The object is detected pixel-wise by subtracting the current image from a reference background image previously obtained. In this approach, there are two major challenges, namely changes in ambient illumination and background pixels whose color changes periodically. Stauffer and Grimson [87] proposed a de facto standard solution to the problems. Fig. 3.2 illustrates the mechanism. The illumination changes are addressed by using an online machine-learning approach. Periodic color changes are addressed by applying a mixture of Gaussian (MoG) model to the pixels. However, the foreground pixels in the MoG models often lose focus, making part of the body of the target object disappear. A CNN has been employed [4] to extract pixels probabilistically, which achieves better foreground definition than a MoG model.

Another background-subtraction approach involves edge subtraction. Both the current and reference video frames are converted to edge lists using a Canny algorithm [6]. The edges of a moving object are obtained by comparing the edge lists. Compared with the MoG model, this model is less sensitive to illumination changes. However, the extracted edges usually change their shapes and positions across successive frames. A statistical approach [75] may solve this problem.

Object recognition based on feature extraction is another approach to vehicle detection. Cascade classifiers based on Haar features [93] are often employed in this approach. Their algorithm comprises two phases, namely region proposal and filtering. In the former phase, all possible subimages are clipped from the original image, with their sizes and positions being varied. Next, in the latter phase, each subimage is tested to determine whether it is a vehicle by AdaBoosting [16] that uses Haar features. In real video analysis, the detector often loses sight of a vehicle for a moment, so a MoG model is utilized as a backup detection system [85].

Recently, a CNN-based detector called an R-CNN has been proposed [18]. The R-CNN also comprises a region proposal step and a classification step. In the proposal step, a selective search algorithm[91] is employed as opposed to using brute-force extraction. This algorithm

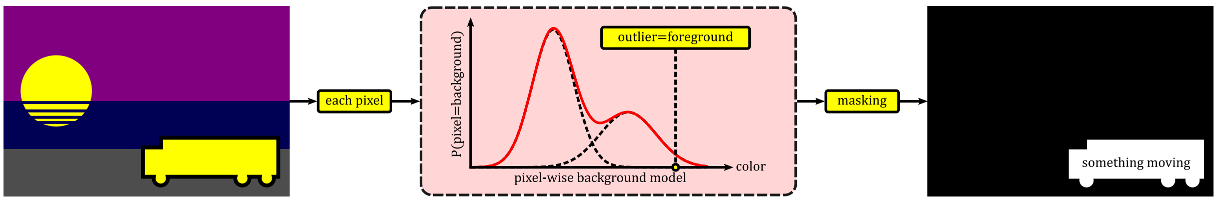


Figure 3.2: Moving object detection via background subtraction.

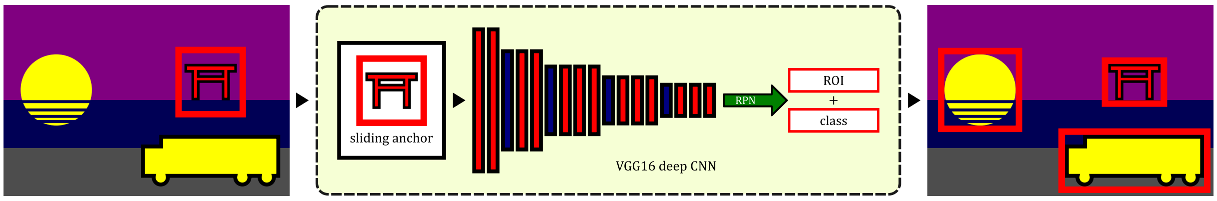


Figure 3.3: Region proposal scheme of Faster R-CNN.

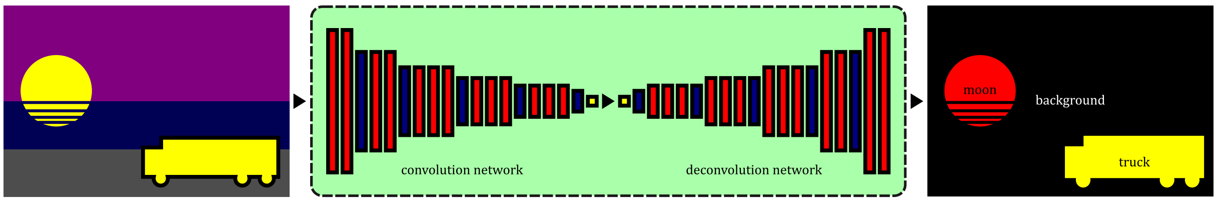


Figure 3.4: Semantic segmentation scheme of a fully convolutional network.

first identifies initial segments via a graph-based method and then groups them in a hierarchical bottom-up fashion. That process dramatically reduces the number of subimages to be processed in the next step, contributing to speeding up the detection. In the classification step, the R-CNN converts subimages into feature vectors and classifies them one by one using support vector machines instead of AdaBoosting. Ren et al. [73] improved the R-CNN by introducing a region proposal network (RPN) instead of the selective search. First, a deep CNN called a VGG16 [84] developed by the visual geometry group (VGG) at University of Oxford extracts features from the image. In this process, the image is downsized to a low resolution by the VGG16 network. Next, the region proposal network (RPN) [73] finds the approximate bounding boxes of the vehicles and wheel axles. Finally, the classification task and region regression task are performed using features for each subimage clipped by region-of-interest pooling. The RPN was implemented in terms of a CNN. By training the networks in the fashion of multitask learning, the revised recognizer (named *Faster* R-CNN) achieved highly accurate results for region extraction. Fig. 3.3 illustrates the scheme of the RPN.

To extract the shapes (pixels) of vehicle bodies, semantic segmentation may be helpful. Semantic segmentation can be considered as pixel-wise still-image classification, and a fully convolutional network (FCN) has been utilized for that purpose [46]. Fig. 3.4 illustrates the mechanism of FCN. FCN has been attracting considerable attention in the context of computer vision. However, an unknown vehicle that is too large to fit the image into a single video frame may often be divided into many false segments, e.g., a chair and a sofa. In addition, FCN, as well as a MoG model, may not resolve the occlusion problem by itself. Although instance segmentation may be a promising solution to the occlusion problem [67, 12], there is still room for improvement with respect to segmentation recall, as reported in these papers. The other problem was processing throughput, which was much slower than Faster R-CNN in our preliminary experiments. Under these circumstances, the semantic segmentation approach is yet to prove a solution that is an alternative to background subtraction and object recognition.

3.4 Techniques for Vehicle Tracking

After a moving vehicle has been detected, it needs to be tracked through successive frames. A graph-based approach that considers the relative positions of moving vehicles has been proposed [83]. An alternative approach [90] detects an identical vehicle in different frames by utilizing histograms of oriented gradients and color histograms, and this approach is robust against low frame rates and low-resolution videos.

Beyond the vehicle tracking context, some methods are generally applicable to a variety of types of objects and environments. In general, such methods utilize two models, namely an appearance model and a dynamic model. Given a video frame, the appearance model suggests several target candidates, whereas the dynamic model removes candidates that are too far from a previous target position. To make the algorithm robust against temporal changes in target shapes, the appearance model should describe a target template in a simple low-dimensional feature space by using a correlation filter in the Fourier domain [24], spatial graph[44], principal component analysis[77], or sparse coding[53, 1].

Recently, it was discovered that a deep CNN could be useful for object tracking [94]. The different layers of a CNN encode features of different granularity, with a deep layer capturing highly abstractive categories and a shallow layer capturing detailed and discriminative features. Moreover, the activated region of a layer accurately captures the shape of the target object. Based on these observations, an online tracking algorithm that involves information from several different CNN layers has been proposed.

In general, these algorithms implicitly assume that the tracked target exists in a wide-angled video for a substantial period, which is not necessarily a valid assumption for traffic analysis on real bridges because of neighborhood privacy protection requirements.

Chapter 4

Traffic Surveillance System

As a demonstration of a minimal CPS, we are conducting some experiments on two bridges in Japan. We have deployed a traffic surveillance camera at an entrance to the bridge and an accelerometer for vibration measurement at one of the spans inside the bridge. By combining the data from these sources, we aim to detect early signs of possible bridge collapse. First, the TSS detects vehicles that pass across the bridge in chronological order. Next, the data acquisition system collects the vibration response at the bridge span. The bridge deterioration can be detected by analyzing the vibration responses of free vibration after a vehicle has passed over the sensor.

Our proposal in Chapter 4 comprises two subsystems, namely a traffic surveillance system (TSS) and a vibration analysis system (VAS). The TSS described in Section 4.3 is based on a deep convolutional neural network (CNN)[73] that can recognize individual passing vehicles. The VAS described in Section 4.5 is only a plan for now, but it can detect changes in vibration characteristics.

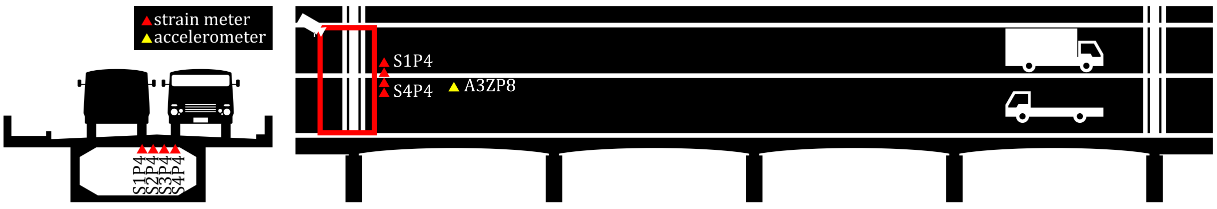
Our main aim is to extract free oscillations from the observed vibration data following events of passing vehicles. In addition, we study the vehicles, e.g., whether they are full trailers or a semitrailers, empty or fully laden, because additional information can be used for other purposes such as detecting overloaded vehicles. From the perspective of the signal vs. noise (SNR) ratio, the extracted oscillations caused by a large, heavy vehicle are preferred. Therefore, our system identifies such vehicles by counting their axles. The camera looks down on a vehicle at such an angle that it can see both its top and one side, enabling the type of vehicle and the number of axles to be assessed.

4.1 Observation Environment

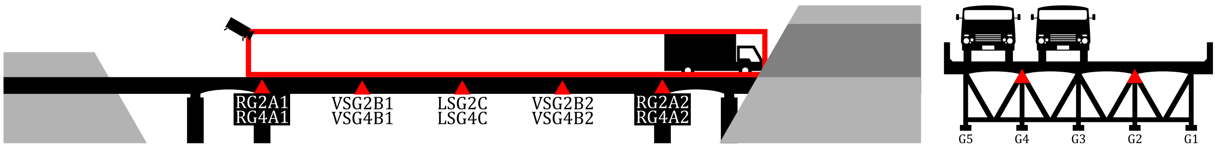
In this dissertation, we demonstrate our proposals for two road bridges in Japan, denoted Bridge C and Bridge S. Bridge C (concrete) is a 300-m prestressed-concrete (PC) bridge with four spans and two lanes, as shown in Fig. 4.1 (a). Bridge S (steel) is a 45-m steel expressway bridge with five girders, five beams, and two lanes, as shown in Fig. 4.1 (b). In later chapters, the denotations refer to the same two bridges.

We deployed strain sensors and accelerometers inside the bridges and a surveillance camera above each bridge. We explain all sensors of the sensing systems in this section, although the strain sensors are not utilized in this chapter but are addressed in later ones. Table 4.1 shows the specifications of the two sensing systems.

On Bridge C, we installed four high-sensitivity strain sensors beneath the deck slab. These are shown as the four red triangles denoted S1P4, S2P4, S3P4, and S4P4, respectively, in Fig. 4.1 (a). The sensor model was the PKM-50S, manufactured by Tokyo Measuring Instruments Laboratory, Co. Ltd., and had four strain gauges and a built-in bridge circuit to



(a) Bridge C.



(b) Bridge S.

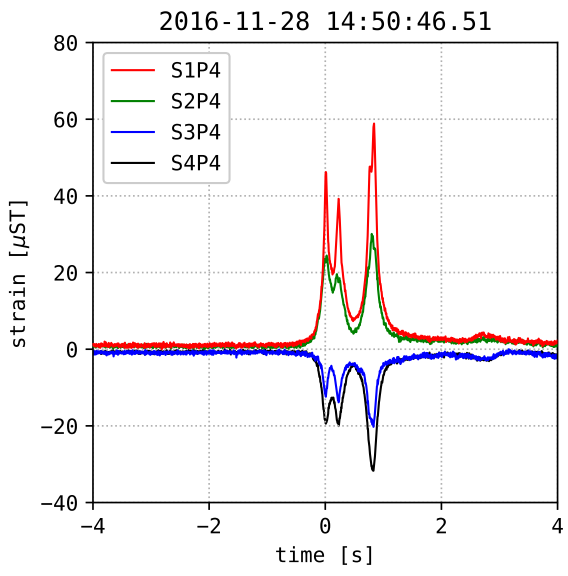
Figure 4.1: Installation positions of strain sensors and accelerometers.

Table 4.1: Comparison of the two observation environments.

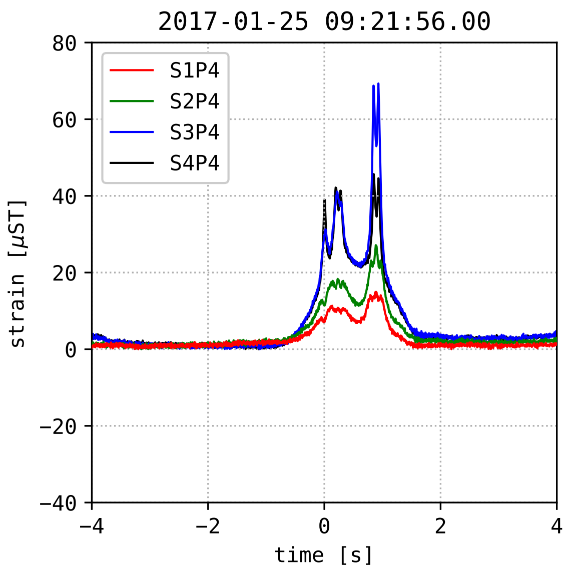
	Camera		Sensor channels		
Bridge	Resolution	Clarity	Strain	Accel.	Rate
C	1280×960	Clear	28 ch	34 ch	200 Hz
S	640×480	Noisy	33 ch	5 ch	100 Hz

achieve a resolution of 1/3, compared with a single-gauge transducer. These sensors collected strain responses at a deck slab in the direction orthogonal to the bridge axis. As seen in Fig. 4.2, the sensors could react to vehicles at a distance of up to 10 m, considering the vehicle speed (about 10–20 m/s) and the wave width. In addition, we used an accelerometer installed at the center of the leftmost span on Bridge C, as shown by the yellow triangle denoted A3ZP8 in Fig. 4.1 (a). The sensor was installed at the bottom of the box girder. The model was the JA-70SA, manufactured by Japan Aviation Electronics Industry, Limited, and had a sensitivity of $0.2039[\mathrm{Vs}^2/\mathrm{m}]$. It should be noted that this accelerometer samples vibration in three axial directions. A3ZP8 samples vertical vibration and the remaining two axes, X (horizontal vibration in the direction of the bridge axis) and Y (horizontal vibration in the orthogonal direction), were denoted A3XP8 and A3YP8, respectively.

The two graphs in Fig. 4.2 show examples of the observed strain signals for Bridge C after a large vehicle enters the bridge from the left to the right (LtoR) and from the right to the left (RtoL). For the strain data, the spikes show the time when the wheels passed over the sensor. Fig. 4.3 shows examples of observed acceleration data at the same times on Bridge C. For the vibration data, the bridge began to vibrate forcibly in response to the moving axle loads. In general, accelerometers tend to capture the small vibrations caused by vehicles distant from the

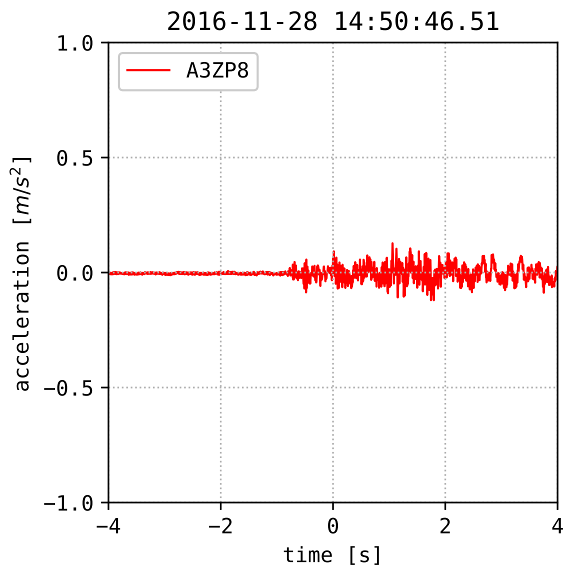


(a) An LtoR vehicle.

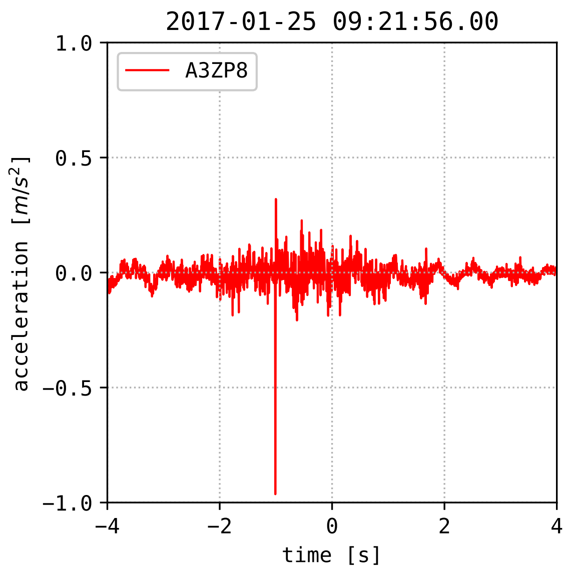


(b) An RtoL vehicle.

Figure 4.2: Examples of strain responses caused by vehicles.



(a) An LtoR vehicle.



(b) An RtoL vehicle.

Figure 4.3: Examples of acceleration responses caused by vehicles.

bridge span and therefore are less suited to vehicle detection. In contrast, a strain sensor can easily identify an individual vehicle.

On Bridge S, we installed three types of strain sensors. Sensors of the first type were placed on vertical stiffeners, denoted VSG2B1, VSG2B2, VSG4B1, and VSG4B2. VSG2B1 and VSG2B2 were installed at the intersection points of the girder G2 and the two beams B1 and

B2. VSG4B1 and VSG4B2 were installed at the intersection points of the girder G4 and the two beams B1 and B2. The second type of strain sensors comprised those installed on lower flanges, denoted LSG2C, LSG4C, and they were installed underneath the two girders G2 and G4. The third type of strain sensors comprised those installed on the bridge supports, denoted RG2A1, RG2A2, RG4A1, and RG4A2. RG2A1 and RG2A2 were installed on the supports of the girder G2, while RG4A1 and RG4A2 were installed on the supports of the girder G4. RG2A1 and RG4A1 were located at the bridge entrance, while RG2A2 and RG4A2 were located at the bridge exit.

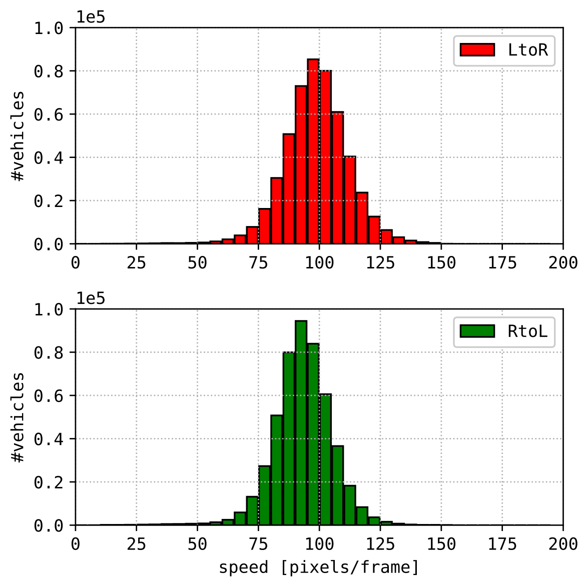
For each bridge, all of the sensors were synchronized except for the camera, and we needed to correct the time shift between the clocks for the sensors and camera. For Bridge C, the sensor clock gained 2.88 seconds per day until November 4, 2016. After November 4, the sensor clock lost 4 seconds per day. The sensor clock was reset at 00:30 every day, and the time shift between the sensors and the camera was cleared to zero seconds. For Bridge S, the sensor and camera were synchronized once at 00:00 on June 19, 2015, and subsequently lost 3 seconds per day. In this dissertation, all timestamps are synchronized to the camera time unless otherwise specified.

4.2 Traffic Dataset Preparation

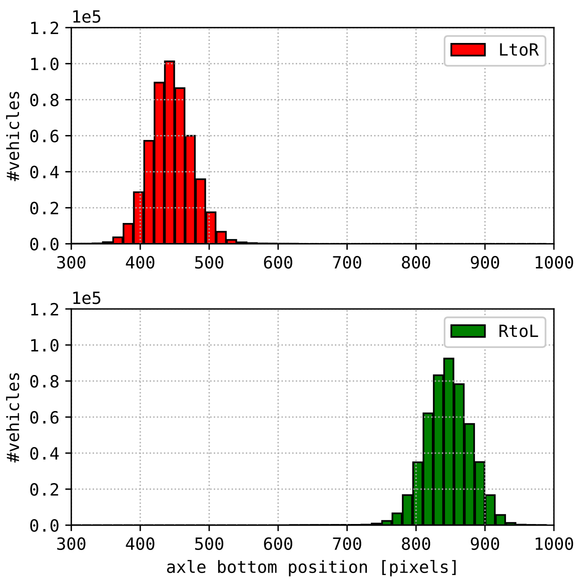
In preparation for the experiments in later chapters, we prepared ground-truth datasets for Bridges C and S, which were denoted DS601 and DS801, respectively. DS601 was created from videos recorded between 08:00 and 16:00 from November 5, 2016, to April 28, 2017. DS601 contained data for the 996,093 vehicles, excluding pedestrians, bicycles, and motorcycles. Fig. 4.4 shows the statistical distributions for vehicle speeds, loci (traveling position in a lane), axle numbers, and appearance intervals. According to Fig. 4.4, there were 502,943 vehicles traveling from LtoR, with the remaining 493,150 vehicles traveling from RtoL. The second dataset, DS801, was created from videos recorded between 06:00 and 18:00 from June 22 to 28, 2015, except for June 26. DS801 contained data for the 92,523 vehicles that crossed Bridge S.

Some vehicles were removed automatically from the traffic datasets to stabilize the training process of the neural networks.

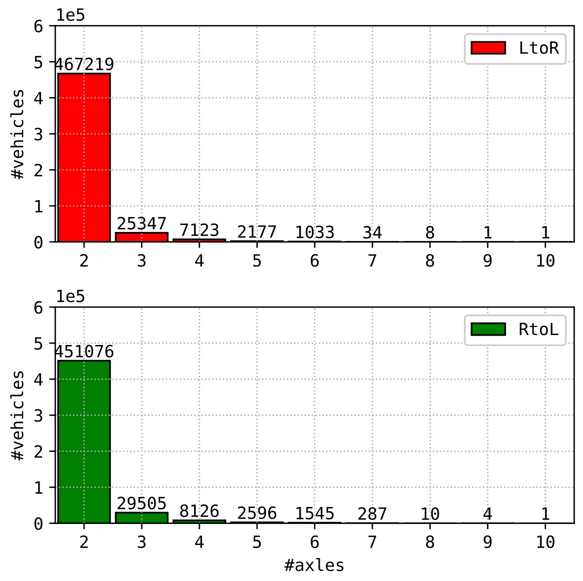
For Bridge C, the following vehicles were removed automatically from DS601. First, vehicles crossing Bridge C in the wrong lane were ignored individually. Second, vehicles that were apparently faster than 82.47 km/h were ignored because these vehicle speeds might be unreliable. It should be noted that the legal speed limit on Bridge C was 60 km/h. Third, vehicles which were apparently slower than 5.5 km/h were also ignored individually because the dynamic response at the target span on Bridge C might then extend beyond the time window used in neural networks in Chapter 5. In addition, video data recorded on the days listed in Table 4.2 were ignored. We ignored video before November 5, 2016, because the



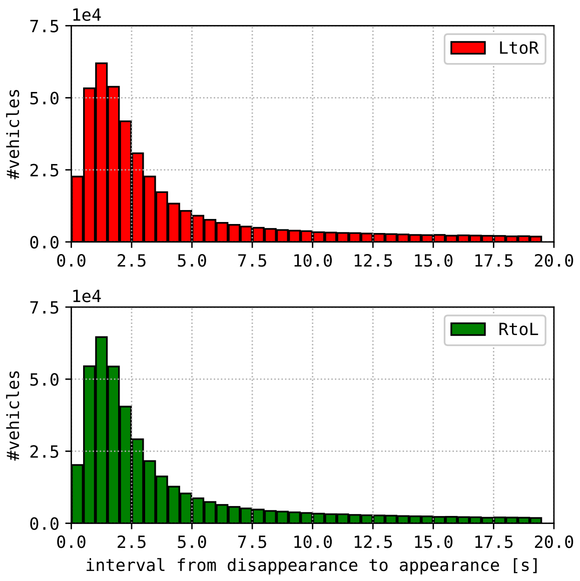
(a) Traveling speed.



(b) Traveling locus.



(c) Number of axles.



(d) Interval moment.

Figure 4.4: Statistical distributions for DS601.

synchronization software for the sensing system was replaced on November 4. We ignored video data on November 9, 2016, because one of the two lanes on the bridge was closed because of construction work. Under such conditions, the bridge behaved abnormally because many vehicles were required to use the open lane in the opposite direction to that normally used. We ignored video data on November 24, 2016, because we had some doubts about the synchronization between the sensors and camera. We ignored video data from January 6 to 15, 2017, because the sensor data during this period was lost because of a fault in the observation

Table 4.2: List of removed days (DS101).

2016-11-01	2016-11-02	2016-11-03	2016-11-04
2016-11-09	2016-11-24	2017-01-06	2017-01-07
2017-01-08	2017-01-09	2017-01-10	2017-01-11
2017-01-12	2017-01-13	2017-01-14	2017-01-15

environment.

For Bridge S, vehicles that were estimated as traveling faster than 224.0 km/h or slower than 11.2 km/h were removed from DS801. Such cases might occur mainly when the strain responses caused by multiple vehicles cannot be separated in terms of the time window being used in the speed estimation process.

4.3 Traffic Surveillance System

Section 4.3 describes two versions of the TSS, one of which was first proposed for natural vibration analysis [32], and the other was an improved version [34] for preparing weatherproof traffic datasets described in Section 4.2.

4.3.1 Original TSS

Fig. 4.5 illustrates the mechanism of TSS. Once a vehicle enters the bridge, the TSS detects *something moving* by using a background-subtraction detector [87]. Then, a CNN-based object detector called a Faster R-CNN [73] finds a bounding box. The Faster R-CNN is robust against occlusion but requires substantial computational time for processing every video frame. The background-subtraction approach is the exact opposite. Although there are differences between the various subtraction models, as discussed in Section 3.3, we simply adopted a MoG model[87], which could be replaced if necessary. In a previous paper [32], we mentioned that the CNN is poorly suited to estimating an accurate bounding box for the vehicle and often loses sight of the target. This problem was resolved in the later version for Section 4.2 by increasing the variation of vehicle images for training the CNN.

Extracted regions are assumed to belong to one or more vehicle images or to contain isolated noise pixels caused by illumination changes. First, the TSS removes the noise regions by setting a maximum number of vehicles and their minimum sizes. Next, a Faster R-CNN[73] subsystem using the VGG16 network[84] classifies the targets and deals with occlusion. The dataset obtained involved many cars under daytime and nighttime conditions being observed from a variety of angles and distances. From the observation of actual videos, we found that

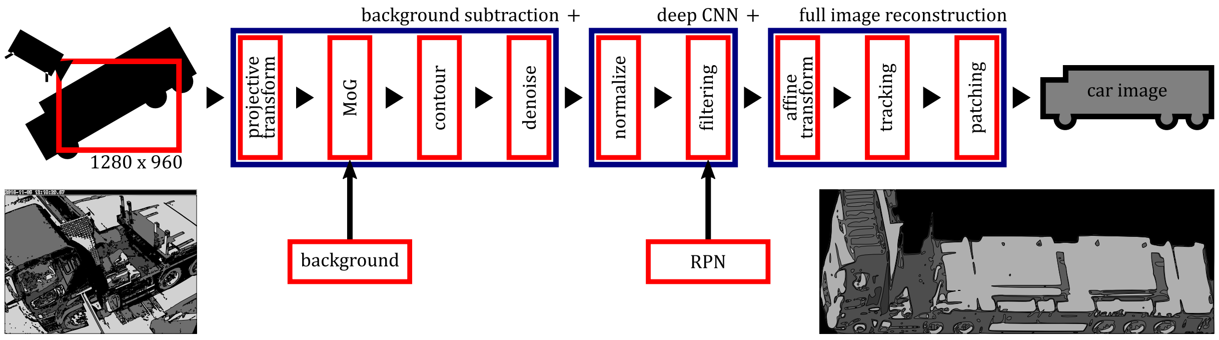


Figure 4.5: Traffic surveillance system (TSS).

a vehicle region suggested by a Faster R-CNN is more accurate for analyzing the underside that includes the axles, whereas the roof region can frequently be ignored. For this reason, the system resolves occlusions not by extracting the target, but rather by removing non-target elements from the region. If no vehicle is detected directly by the Faster R-CNN but there is a large moving object, the TSS assumes that the largest such object represents a vehicle. Such a case usually occurs when a large trailer appears after its tractor has gone out of frame; this case occurs less often with small cars. A trailer can conceal anything behind it, so this heuristic assumption is valid in practice. This problem was partially resolved in the later version for Section 4.2 by telling the CNN that such objects are parts of vehicle bodies.

We employed a collision detection approach for tracking, because the video's frame rate should be set high enough to enable a detailed investigation of the vehicle body, including Chapter 6. After a vehicle has passed out of the field of view, the TSS catenates the previous frames. First, the TSS performs a projective transformation to make the vehicle's images in the previous frames appear horizontal. Next, the TSS clips out subimages from the frame's center by using a clipping window whose width is determined by the vehicle's speed, and the cutout images are placed in order. Finally, the TSS performs an affine transformation to smooth the joint face between patches, and the perspective difference between the vehicle's underside and roof is absorbed.

A TSS can identify moments when only one large vehicle has crossed the bridge and no other large vehicles are involved. A large vehicle generates a large vibration for an extended period before and after it runs over the vibration sensor. This extended period of vibration may negatively affect the vibration analysis of another vehicle passing immediately after the large vehicle, particularly if the two vehicles run over the vibration sensor at the same time. Our system can deal with these issues by identifying vehicles by axle counting. First, the TSS detects the positions of the wheel axles underneath a vehicle tracked by the TSS by utilizing the Faster R-CNN. The TSS lists the axle positions in chronological order. The axles are then grouped into axle clusters by using the DBSCAN [15] algorithm and considering the vehicle's

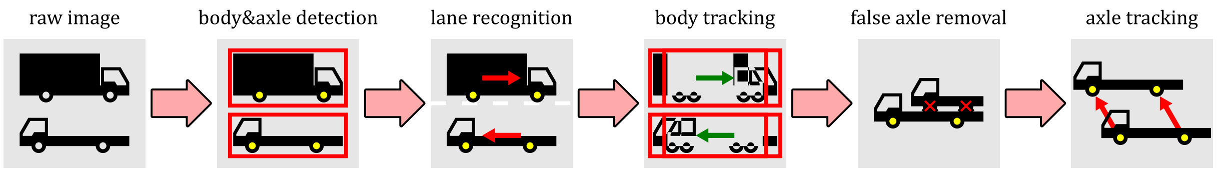


Figure 4.6: Evolved traffic surveillance system (ETSS).

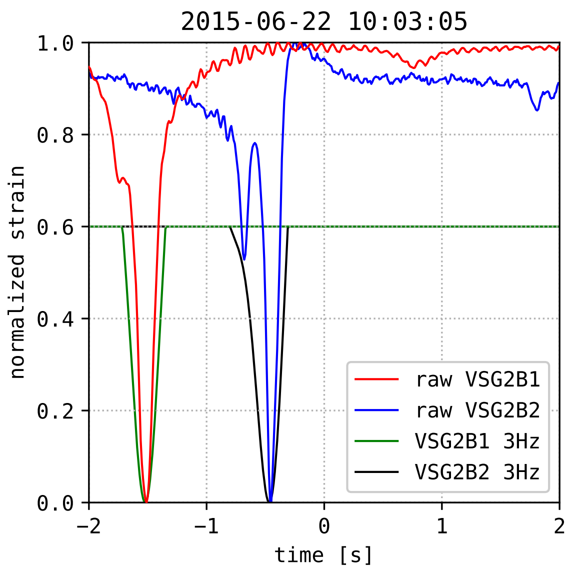
speed. Finally, the TSS checks if the vehicle *looks heavy* based on the number of wheel axles.

4.3.2 Evolved TSS

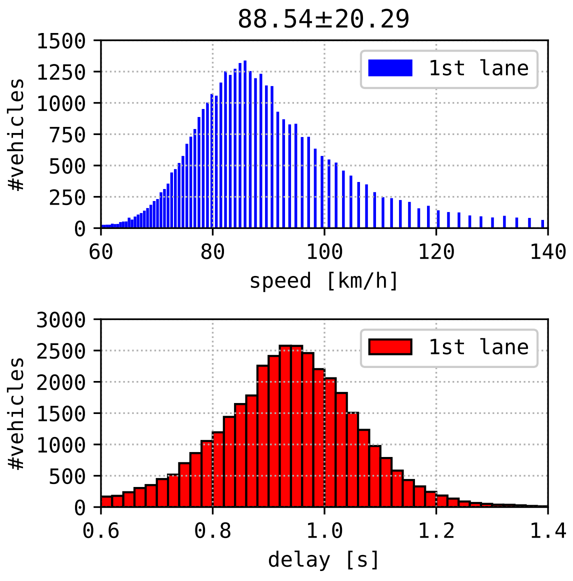
The traffic datasets described in Section 4.2 were created via an improved version of the TSS, as shown in Fig. 4.6. The surveillance cameras installed at the entrances to Bridges C and S captured individual vehicles entering or leaving the bridges. The TSS then outputs the vehicle properties, including time, lane, speed, locus, and number of axles. This process is achieved as follows. After a vehicle enters the bridge, the Faster R-CNN [73] detects a bounding box for it. For Bridge C, the Faster R-CNN was trained with 13,407 images named DS702 that included original VOC2007 [66] images and an additional 3444 vehicle images. For Bridge S, the Faster R-CNN was trained with 15,087 images named DS703 with an additional 5124 vehicle images. Each bounding box is classified into one of the two lanes by considering its bottom position. The TSS then starts tracking the vehicle body over consecutive video frames. In this process, the TSS calculates the approximate speed for the vehicle. The Faster R-CNN also identifies the wheels. After linking wheels to the nearest vehicles, false axles, which can be seen frequently with car carriers, are removed. Finally, the vehicle speed, locus, and number of axles are estimated by tracking each axle.

It should be noted that the coordinates of the wheel images are transformed so the axle appears to travel horizontally. Consequently, the pixel distance in the transformed image is proportional to the distance traveled in meters. For DS601, the ratio of distance in meters to that in pixels was determined as 6.11 mm/pixel and 8.14 mm/pixel for horizontal and vertical directions, respectively, by watching two vehicles with known wheelbases via the camera. We ignored lens distortion because it is not remarkable on either bridge.

For Bridge S, we obtained ground-truth speeds by using the two strain sensors, VSG2B1 and VSG2B2, as shown in Fig. 4.1 (b). The TSS was unable to output the speed, locus, and number of axles because the Faster R-CNN could barely detect the axles because of significant blurring (i.e., noise) in the video signal. We therefore abandoned the TSS for the speed estimation on Bridge S and developed a speed estimator based on peak-signal detection. Fig. 4.7 (a) shows the actual strain responses sampled by the two strain sensors when a vehicle enters Bridge S. The distance between the two sensors was 22.4 m. First, we applied a 1 Hz low-pass filter (LPF)



(a) Peak detection.



(b) Detected speed.

Figure 4.7: Speed estimation mechanism for DS801.

to the raw signals and extracted a 4 s signal for each vehicle. Each signal was normalized and clipped, as illustrated in Fig. 4.7 (a). We then obtained the phase difference between the two signals by calculating the signal correlation. Fig. 4.7 (b) shows the distribution of the estimated vehicle speeds.

This development was unfortunate but had the side benefit of enabling us to demonstrate two scenarios for the preparation of ground-truth data. In the first scenario, we can extract rich information from the traffic surveillance video and do not have to install many additional sensors. In the second scenario, the video quality may be poor, but we can install multiple strain sensors for vehicle detection during the system-optimization period. The installation point of the surveillance camera may sometimes be restricted because of privacy considerations, such as the camera pointing directly at nearby houses, and the second scenario can then be an alternative approach. A discussion of how to utilize additional sensors for axle counting and locus estimation in the second scenario is outside the scope of this dissertation.

4.4 Experimental Results

We tested the functions of the TSS, namely vehicle detection and axle counting, at Bridge C in Japan. The camera was installed on a pole mast located at the southern entrance to the bridge. The camera looked down on a vehicle at such an angle that it could see both its top and one side, and the camera was utilized to identify the type of vehicle. The bridge had two opposing

Table 4.3: Processing time and accuracy.

Recording date	Time	TP	FP	PR	GT
10-24 00:00–30 ☁	36’22	41	31	56.9%	18
10-24 06:00–30 ☁	28’56	39	9	81.2%	40
10-24 12:00–30 ☁	45’37	178	18	90.8%	192
10-24 16:00–30 ☁	45’04	208	8	96.3%	215
11-02 00:00–30 ☼	27’49	11	14	44.0%	12
11-02 06:00–30 ☼	28’40	42	2	95.5%	43
11-02 12:00–30 ☼	42’47	183	16	92.0%	191
11-02 16:00–30 ☼	46’56	215	11	95.1%	215
11-05 00:00–30 ☼	33’20	16	54	22.9%	15
11-05 06:00–30 ☼	29’54	57	1	98.3%	57
11-05 12:00–30 ☼	54’06	301	1	99.7%	315
11-05 16:00–30 ☼	47’00	253	1	99.6%	256
11-28 00:00–30 🌧	27’14	36	0	100.%	10
11-28 06:00–30 🌧	35’08	48	64	42.9%	42
11-28 12:00–30 🌧	35’06	112	39	74.2%	110
11-28 16:00–30 🌧	38’30	147	24	86.0%	151
12-10 12:00–30 ❄	41’03	144	14	91.1%	147

lanes, but the camera was aimed at the southbound (nearside) lane. The camera was set at 1280×960 resolution and 25 FPS, and the videos were downscaled into 640×480 resolution during processing.

4.4.1 Original TSS

We conducted experiments with real videos recorded on October 24; November 2, 5, 28; and December 10, 2016. We investigated whether the system could detect vehicles properly and whether it could distinguish large trucks accurately from other vehicles. The tests were held under a range of conditions that included daytime and nighttime, and sunny, cloudy, rainy, or snowy weather. We were able to identify situations in which the system had difficulty detecting vehicles.

For vehicle detection, we utilized a Faster R-CNN that was trained for the VOC2007 [66] dataset.

Table 4.3 shows the processing time for the videos using an NVIDIA Tesla P100 GPU. On average, 19.8 frames per second can be processed, which is regarded as almost real time. The TSS slows down remarkably while tracking a vehicle, with the slowdown being caused mainly by forward propagation in the Faster R-CNN. The processing time may therefore degrade with increasing traffic volume, for which load balancing among GPUs could be a solution.

Table 4.3 also shows the precision rate (PR) with the numbers of true-positive (TP), false-positive (FP), and ground-truth (GT) vehicles. These scores were obtained by comparing the detection results with an annotation dataset DS501 prepared by hand to validate the accuracy. The results indicate that the performance is highly accurate on sunny days but degrades drastically at dawn (or dusk), at night, and during snowstorms. The reason that the TP tends to be much higher than the GT at night is that the vehicle is often misidentified because of tracking failure. At night, although the MoG subtractor can detect movement in the dark, the truth vehicle is difficult to identify. This difficulty is because the road surface reflects the beams from vehicles, and the reflection can appear to be a moving object preceding an approaching vehicle. The details of the vehicle itself are blurred, noisy, and nearly invisible because of the supersensitive imaging. Nevertheless, the positions of headlights and taillights can be specified accurately, and the TSS may be able to capture the approximate positions of vehicles in the future by training the recognizer for nighttime vision. For the snowstorm case, because the video's contrast can be quite low in a snowy scene, a dark-colored vehicle is more likely to be detected than a bright-colored vehicle. The TSS reconstructed 144 TP vehicles in total, but about 40% of the vehicles were malformed, which is far from practical.

Robustness against occlusion mainly depends on the performance of the Faster R-CNN because the MoG model cannot separate *touching* vehicle images by itself. There are two cases involving occlusion, namely 1) a small vehicle in the nearside lane and another vehicle behind and 2) a large vehicle in the nearside lane and another vehicle behind. For case 1), the axles of the rear vehicle will be visible, and both vehicles are likely to be detected as different vehicles, with the occlusion being solved. For case 2), the rear vehicle may be hidden behind the large vehicle in front, and the Faster R-CNN should split the two vehicles at the beginning and the end of the occlusion period. If the TSS fails to segment properly, the reconstructed image may include excrescences in both cases.

For counting wheel axles, we utilized a Faster R-CNN trained for a specialized dataset DS701 that comprised 445 images captured randomly around noon on the sunny days of October 27 and November 2,4, and 5, 2016. We did not analyze morning and evening videos on rainy and snowy days since it would be difficult to collect ground-truth data from the noisy videos. The other conditions were as described in Section 4.3. Note that a signboard at the bottom of the image field could distract the axle detector when an axle appeared behind it. The system was therefore configured to ignore such axles.

Table 4.4 shows the accuracy of the axle-number detection. The sum of the True and False cases in each row equals the GT in Table 4.3. During daylight hours on a sunny or cloudy day,

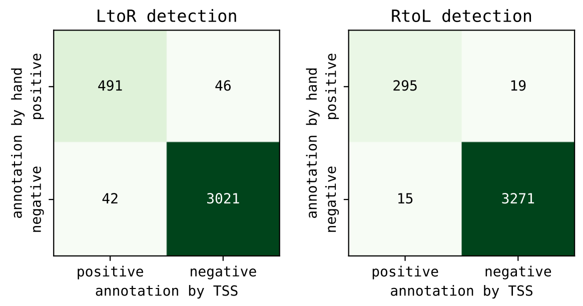
Table 4.4: Accuracy of axle-number detection.

Recording date	True	False	Accuracy
10-24 12:00–30 ☁	163	13	92.61 %
11-02 06:00–30 ☼	16	26	38.10 %
11-02 10:00–30 ☼	205	27	88.36 %
11-02 12:00–30 ☼	168	15	91.80 %
11-02 14:00–30 ☼	237	16	93.68 %
11-02 16:00–30 ☼	196	19	91.16 %
11-05 06:00–30 ☼	20	37	35.09 %
11-05 10:00–30 ☼	235	69	77.30 %
11-05 12:00–30 ☼	285	16	94.68 %
11-05 14:00–30 ☼	315	24	92.92 %
11-05 16:00–30 ☼	245	8	96.84 %
11-28 12:00–30 🌧	33	79	29.46 %
12-10 12:00–30 ❄	7	137	4.86 %

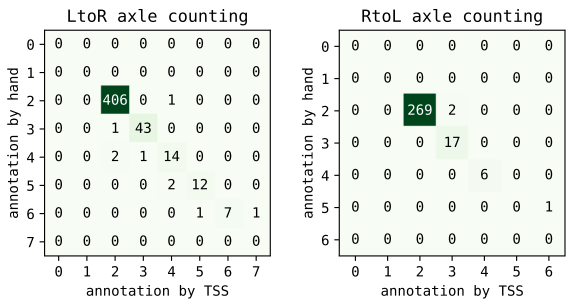
the system counted axles with 90% accuracy. However, the accuracy was reduced at dawn, or on rainy or snowy days, because of the darkness. It might be better to postpone running the system in such cases by considering the weather forecast for the day, because the system's aim is to monitor potential damage over a period, and one or more vibration samples per day may be sufficient for such a purpose. Consequently, we focus on precision rather than recall, and the TSS performance given in Table 4.3 is adequate. Errors in the TSS and the VAS may distort the signals of damage detection, which may be compensated by collecting many samples.

4.4.2 Evolved TSS

Because the datasets introduced in Section 4.2 were created fully automatically, we also prepared an annotation dataset DS201 by hand to validate the accuracy of DS601. Fig. 4.8 shows a comparison of DS601 and DS201. Fig. 4.8 (a) and Fig. 4.8 (b) show confusion matrices for the tasks of vehicle detection (Task 1) and axle counting (Task 4), respectively (defined later in Table 5.1). By focusing on the sums of true-positive cells on the diagonal lines, the dataset achieved moderate results for precision and recall. It should be noted that the term *vehicle detection* refers to each vehicle entering or leaving the bridge and is evaluated every second. Therefore, the two tasks can be treated as simple classification tasks. DS201 was created from 60 videos that were randomly extracted from the half-year video records for Bridge C. Each video was of 60 s duration and recorded under a representative variety of weather conditions that included sunshine, heavy rain, snow, and snow accumulation. Although



(a) Task 1.



(b) Task 4.

Figure 4.8: Confusion matrices for accuracy validation of DS601.

there were many situations poorly suited to video analysis, the TSS achieved high detection accuracy for Bridge C. Unfortunately, the tasks of speed estimation and locus estimation were too difficult to annotate by hand. However, wrong estimation of speed and locus may lead to wrong estimation of axle numbers, and we are confident about the accuracy of speed and locus because we confirmed the accuracy of axle counting.

4.5 Vibration Analysis System

Fig. 4.9 illustrates the second component of our proposed system, the VAS, which extracts a vibration response after a vehicle passes and fits Eq. (2.3). The VAS exploits the TSS for vehicle detection because the TSS can identify moments when a vehicle has passed and no other vehicle is stressing the bridge, meaning that the bridge vibrates in a manner of free oscillation. Such moments are difficult to identify by vibration analysis alone, especially when the vibration characteristics become deteriorated because of damage.

The variation in the vibration response attributable to environmental conditions such as the season, the weather, and the temperature is an issue. Such a problem has been considered previously [11, 50, 29] in the context of classical natural-frequency-based damage detection [2] that does not involve time constants. Our task was to characterize relations among the frequencies, time constants, and environmental variables using mining approaches. We could then identify *abnormal* vibrations by anomaly detection. This method might be a more appropriate approach to detecting damage that could be applied more generally than the naïve approaches based on natural frequency.

We demonstrate the mixture model defined in Eq. (2.3) using the accelerometer A3ZP8 for Bridge C. First, by using a short-time Fourier transformation, we obtained the natural frequencies below 5 Hz. Next, we extracted the free damped vibrations, as shown in the upper graph in Fig. 4.10 (a), as sampled by the acceleration sensor. The black area indicates the raw

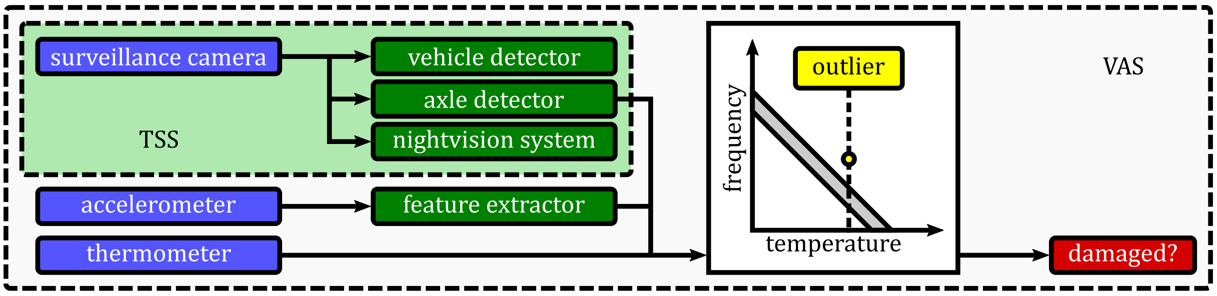
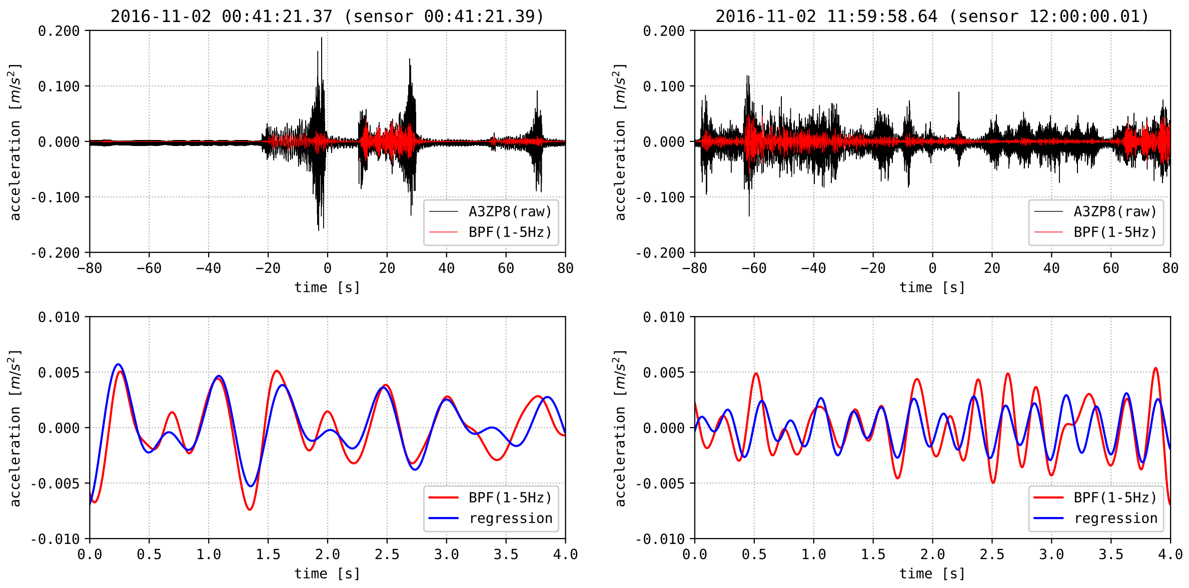


Figure 4.9: Vibration analysis system (VAS).



(a) Light traffic.

(b) Heavy traffic.

Figure 4.10: Exponential fitting to damped oscillations.

vibration sampled at a rate of 200 Hz. The red area shows the vibration smoothed by a 1–5 Hz finite-impulse-response bandpass filter (BPF). Finally, we fitted a mixture of two exponentials to the data, as shown in the lower graph in Fig. 4.10 (a). We employed the matrix pencil method[26, 79] for fitting operations. To exploit the time constants, we needed to remove some forced vibrations caused by the vehicles, which is a difficult problem, particularly in heavy traffic. It should be noted that the TSS was not utilized for the experiments in this section.

To exploit both natural frequencies and time constants as features representing bridge damage, we need to extract free damped vibrations from the actual accelerometer data, removing the forced vibrations caused by the vehicles. However, the extraction is not easy, especially while many vehicles cross the bridge incessantly. Fig. 4.10 (b) shows such a case. The best solution is to close the bridge temporarily and let a heavy truck cross the bridge for testing, but considering the number of bridges to be monitored, that is not a reasonable solution.

4.6 Discussion

On Bridge C, the difficulty was the narrow-angled camera. In many cases, the camera's angle is limited because of neighborhood privacy protection requirements. A trailer truck can be so large that a single camera cannot fit the image into a single video frame except when installing the camera well away from the bridge. Therefore, we may have to catenate a number of frames to reconstruct the overall image of the truck, and for accurate concatenation, we need to locate the exact position of the truck in each frame. This task can be challenging because a feature point used in the tracking can appear or disappear suddenly during the video. Therefore, tracking large trucks can be difficult, and feature-based tracking methods cannot necessarily accommodate the disappearing-feature problem by themselves. They frequently lose sight of the vehicle even if it is at the very center of the image field. Such a condition was noticed particularly when tracking a large vehicle, and we concluded that the recognizer is most likely to make this mistake when most of the target's body is not within the image field. Consequently, we need to employ a traditional background-subtraction approach. However, this cannot solve occlusions by itself, so we used a deep CNN as a solution. This problem was partially resolved in the ETSS in Section 4.3.2 by adding such images to DS702 and DS703, which were used as training data.

4.7 Conclusion

In this chapter, we have proposed a monitoring system that detects large vehicles utilizing background subtraction and a deep CNN. In addition, we have proposed a vibration-monitoring system that exploits natural frequencies and their time constants by extracting free vibrations. Both proposals should be suitable for monitoring a bridge's accumulated damage with a day-to-day granularity.

Three tasks need to be urgently addressed. The first issue of current concern is poor detection at night, which we may overcome by introducing a night-vision system. This system would exploit a vehicle recognizer trained for real nighttime videos in particular and focus on the position of headlights and taillights. The second task is to adopt state-of-the-art outcomes from CNN research, such as semantic and instance segmentation, to enable the TSS to more accurately extract vehicle shapes from videos. In the work reported in Chapter 4, we adopted a temporary solution, namely the MoG model, to satisfy the requirement for real-time processing. However, we will be able to exploit segmentation-based vehicle extraction when it offers real-time processing in the future. The third most pressing matter is that a single bridge's vibration data before and after the bridge becomes damaged are insufficient. We shall therefore seek to set up the systems on many bridges other than Bridge C. We would like to substantiate the effectiveness of the mixture model, which exploits both the natural frequencies and their time constants, as soon as possible.

Chapter 5
Vehicle Detection from Sensor Data

Chapter 5 introduces a data-driven vehicle detection system for a BWIM system consulting only a single strain sensor.

Many BWIM studies [65, 81, 107, 30, 23] have neglected the rich information on the target vehicles available from an individual sensor. For example, both the strain response and the vibration response can be predicted from a vehicle's properties such as its speed, trajectory, and axle positions. There is therefore the possibility that such properties can be obtained inversely from the signal waveform, particularly if a high sampling frequency is used.

The proposed system is based on a deep CNN, which detects moving vehicles and estimates their properties, taking as input the raw strain signal sequence accompanying each target vehicle's traversal of the bridge. The sequence contains a large amount of rich information in itself, especially when the strain data are sampled at high frequency. For decades, this information has been discarded, but a CNN can exploit it without having to use multiple sensors installed at multiple positions on the bridge. We call this a *deep sensing* approach. This approach may enlarge the role of actual sensors installed on the bridge and may bring two major benefits. First, we may replace some specialized sensors by a versatile, miniature, and inexpensive sensor, potentially reducing the necessity of developing specialized sensing devices. Second, we may simplify entire sensing systems deployed in the real world by using the same single sensor for multiple purposes and by reducing the total number of required sensing devices. Deep sensing may be an antithesis of conventional sensor-fusion approaches that must synchronize many sensors accurately and are therefore unreliable and require frequent inspection and repair.

Because our system uses a single strain sensor for vehicle detection, our proposal can merge the main weighing operation, which also requires a single strain sensor, with the vehicle detection operation. Fig. 5.1 shows the proposed system, named the automobile weighing system (AWS), that uses a subsystem for vehicle detection named the automobile identification system (AIS). Unfortunately, the AWS employs the traditional algorithm [47] for weighing, and the CNN contributes only to the vehicle detection process. The training data can be obtained either from a surveillance camera or from additional strain sensors installed at multiple positions across the bridge. After the CNN is optimized, the camera or additional sensors that are no longer needed can be removed.

Recently, some researchers [81] have developed alternative accelerometer-based BWIM systems instead of using a strain sensor, because a strain sensor tends to peel off from the bridge material, thereby requiring regular inspection. Based on these findings, we have applied the deep sensing proposal not only to strain signals but also to acceleration signals. As we demonstrate in Section 5.3, our proposed system can achieve high detection accuracy for both strain-sensor-based and accelerometer-based systems.

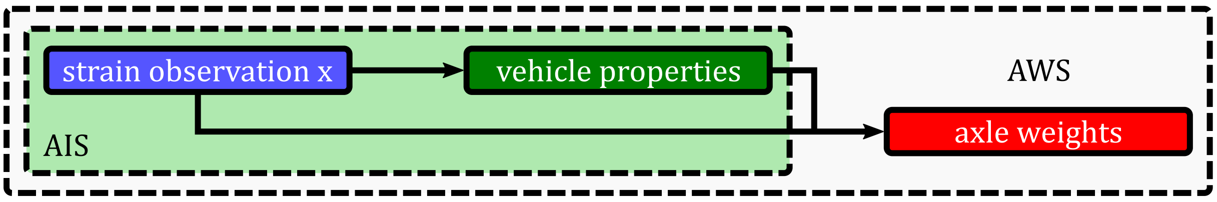


Figure 5.1: System architecture for the proposed BWIM system based on deep sensing.

The contributions of this research are as follows. First, we propose a sensor-fusion framework for collecting training and validation data from real measurements automatically. Second, we obtain vehicle properties of moderate accuracy using only a single strain meter or accelerometer. Finally, we successfully detect vehicles on two kinds of bridges, namely a steel bridge and a PC bridge. Of these, the vehicle detection capability for PC bridges is a major achievement in the world of bridge engineering.

5.1 Proposed Models

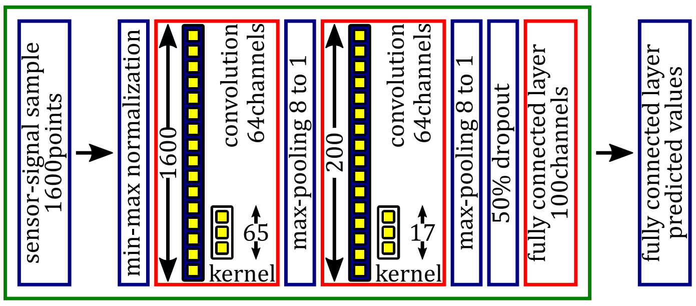
In this section, we explain our deep learning approach to single-sensor data mining. Model parameters are optimized statistically via back propagation [78], by considering actual traffic conditions on a particular bridge. This optimization helps reduce the requirement for heuristic-based axle detection methods, such as naïve peak detection.

We designed three derivative CNN architectures [34] aimed at vehicle detection, and these are shown in Fig. 5.2. The models share the same signature, i.e., they accept a signal sequence for a few seconds as the input and output an estimation result. Among the models, Model 1 is the simplest model, having just two convolutional layers. By inserting a preactivated residual block [22] after each convolutional layer, we improved this model to realize Model 2. Model 3 is the deepest model, with 11 convolutional layers which include five residual blocks. Note that we applied a *third* activation function to the output of the residual blocks in addition to the two activation functions inside the blocks.

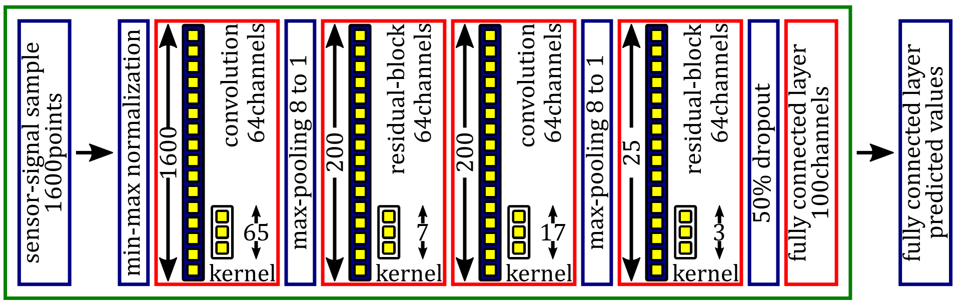
We employed the rectified linear unit (ReLU) [19] defined in Eq. (5.1) for the activation function, except for the final (second) linear layer.

$$f(x) = \begin{cases} x & \text{if } x \geq 0, \\ 0 & \text{if } x \leq 0. \end{cases} \tag{5.1}$$

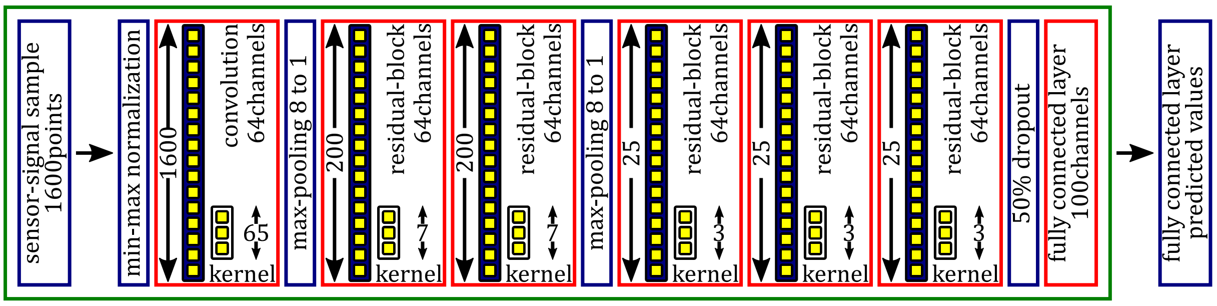
As shown in Table 5.1, the output activation functions for the final linear layer and loss function are defined according to the estimation task involved, i.e., vehicle detection (Task 1), estimating speed (Task 2), estimating locus (Task 3), or counting wheel axles (Task 4). We inserted a 50% dropout [86] between the last convolutional layer and the first linear layer to suppress overfitting.



(a) Model 1.



(b) Model 2.



(c) Model 3.

Figure 5.2: Network architectures for the proposed system.

We also inserted a 20% dropout between the two convolutional layers in each residual block for vehicle detection on Bridge S.

In the world of signal processing, waveform preprocessing (feature extraction) requires care. A typical preprocessing method [103] is to apply multiresolution wavelet transforms to the raw waveform before propagation. Zhang et al. [104] proposed a unique image-based approach that transforms the raw waveform into wave shapes. We adopted the raw waveform approach proposed by Dai et al. [13], who fed the raw audio signal to a very deep residual network. For both strain and vibration data, the signal should be normalized in advance for learning to be effective. Each sequence was rescaled so the maximum and minimum of the sequence were normalized to 1 and 0, respectively.

The four prediction tasks were given to all three networks, differing only in the final linear layer. We could therefore exploit a multitask learning (MTL) technique [9], which may be

Table 5.1: Prediction tasks.

Name	Definition	Output	Activator	Loss
Task 1	Vehicle detection	Existence or not	Sigmoid	Cross entropy
Task 2	Speed estimation	Traveling speed	Identity	Mean squared error
Task 3	Locus estimation	Traveling locus	Identity	Mean squared error
Task 4	Axle counting	Number of axles	Softmax	Cross entropy

beneficial for preventing a neural network from overfitting. We defined the multitask loss $\mathcal{L}$ for N samples as follows.

$$\mathcal{L} = \frac{1}{N}\sum_{k=1}^{4}\lambda_k\sum_{k=1}^{N}\mathcal{L}_k(\boldsymbol{y}_{kn}) \quad \begin{cases} \lambda_1 = 0.2 \\ \lambda_2 = 1 \\ \lambda_3 = 1 \\ \lambda_4 = 0.2, \end{cases} \tag{5.2}$$

where $\mathcal{L}_k$ is the loss for Task k, λ_k is a weight of Task k, and $\boldsymbol{y}_{kn}$ is the output for the n-th sample. It should be noted that the squared losses $\mathcal{L}_2$ and $\mathcal{L}_3$ for Tasks 2 and 3 were divided by the variances of the *trainval* datasets described in Section 5.2. For Bridge S, the multitask loss $\mathcal{L}$ was defined as the follows.

$$\mathcal{L} = \frac{1}{N}\sum_{k=1}^{2}\lambda_k\sum_{k=1}^{N}\mathcal{L}_k(\boldsymbol{y}_{kn}) \quad \begin{cases} \lambda_1 = 1, \\ \lambda_2 = 1. \end{cases} \tag{5.3}$$

By adopting the MTL technique, the networks should be more likely to arrive at a solution common to the four tasks.

5.2 Training and Evaluation Data

We employed the cross-validation approach [40] to validate the generalizability of the three CNNs. Before training, each dataset was shuffled at random and divided into five subsets. One subset was handled as *evaluation* data, with the remaining four subsets being merged and treated as *trainval* data. Unless otherwise specified, 90% of the *trainval* data were assigned as *training* data, with the remaining 10% being *validation* data. The validation data were utilized for early stopping [98]. The evaluation processes were performed with models that updated the minimum of the validation loss for each task. The training and evaluation processes were repeated, swapping the datasets five times and using one subset as the test data in each trial.

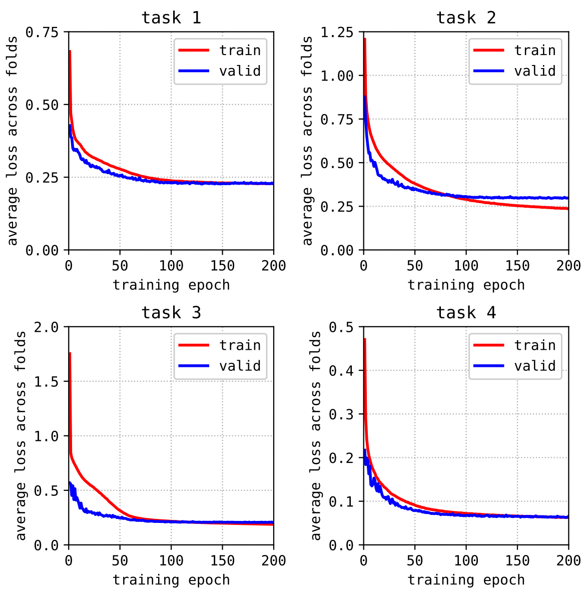
For Bridge C, we prepared four datasets for the supervised training of the four estimation tasks, from DS601 prepared in Chapter 4. The n-th record for Task k is described as a pair

$\{\boldsymbol{x}_{kn}, g_{kn}\}$, where $\boldsymbol{x}_{kn}$ is a sequence of sensor data and g_{kn} is the ground truth for the target variable. For Task 1, g indicates the presence of a vehicle at the bridge entrance at each second. For the other three tasks, g indicates the properties of the individual target vehicles. For LtoR vehicles, $\boldsymbol{x}$ is a raw signal sequence that starts at the same time as the target period. For RtoL vehicles, $\boldsymbol{x}$ is a sequence that ends at the same time as the target period. The length of $\boldsymbol{x}$ was set as 1600 (8 s), as determined by the average speeds and bridge length. It should be noted that the timestamps for Tasks 2, 3, and 4 were shifted deliberately at random within a range of 1 s to prevent the neural networks from estimating the vehicle speeds naïvely from the positions of the strain peaks. These datasets were created individually for the four strain sensors S1P4, S2P4, S3P4, and S4P4, and accelerometers A3XP8, A3YP8, and A3ZP8. Although Bridge C had many strain meters (channels), only a few of the sensors were useful for vehicle detection, because strain is a spatially localized feature, and the utility of a strain sensor can be limited by its installation location.

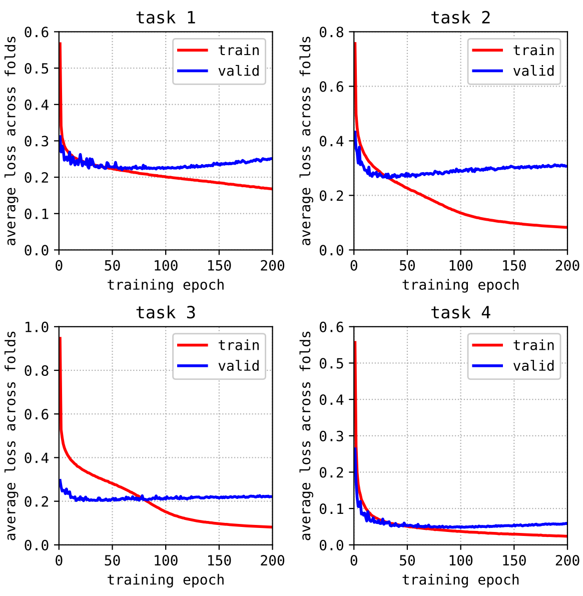
The vehicle detection dataset for Bridge C involved either all LtoR vehicles or all RtoL vehicles, with vehicles traveling in the opposite lane being ignored. By detecting LtoR vehicles but not RtoL ones, or by detecting RtoL vehicles but not LtoR ones, Task 1 could demonstrate lane detectability. The datasets for Tasks 2, 3, and 4 also targeted only LtoR vehicles or only RtoL vehicles. For the strain data, we used vehicles recorded during the leading 30 days in DS601 because of limited time for the experiment. For the vibration data, we used vehicles that had more than two wheel axles because the smaller vibrations caused by passenger cars could be hidden in noise and because bridge engineers are interested mainly in heavy vehicles. Instead, we increased the dataset volume as much as possible, using vehicles recorded during all 163 days in DS601.

For vehicle detection in Task 1, care is needed in the balance of positive and negative samples. To reduce the computation time of an epoch and to improve the quality of the resulting models, we adopted the negative sampling approach [54]. First, we subsampled negative data so that the numbers of positive and negative samples were equal. We then subsampled the detection data again for Task 1 so that the dataset had the same volume as the volumes for other tasks. Because the subsampling may cause significant bias with naturally occurring traffic, this operation was not performed on the evaluation dataset.

For Bridge S, we prepared two datasets for Tasks 1 and 2 from DS801 prepared in Chapter 4. The n-th record for Task k is described as a pair $\{\boldsymbol{x}_{kn}, g_{kn}\}$. For Task 1, g indicates the presence of a vehicle at the bridge entrance at each second. For Task 2, g indicates the speed of a vehicle entering the bridge. For both tasks, $\boldsymbol{x}$ is a sequence of length 400 (4 s) that starts at the same time as the target period. These datasets were created individually for the three types of strain sensors described in Section 4.1.

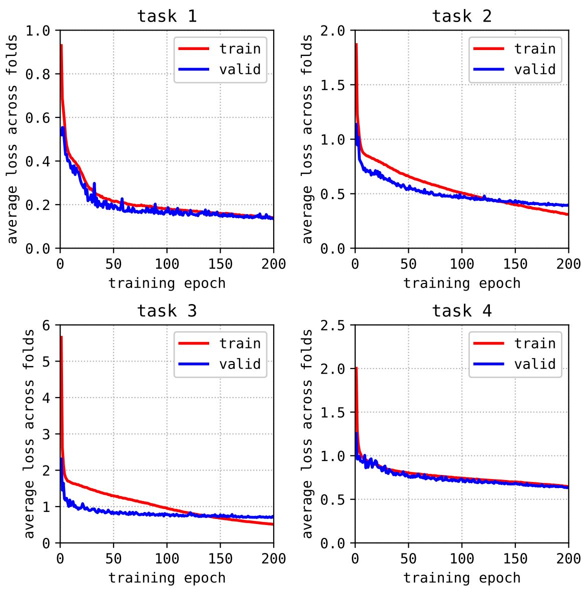


(a) Model1.

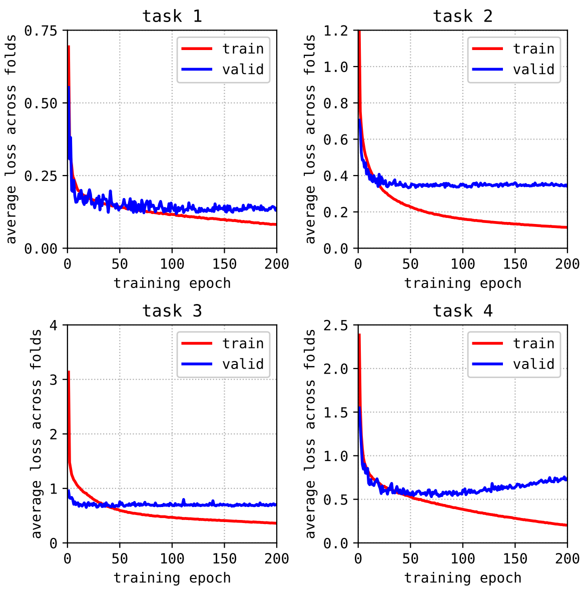


(b) Model3.

Figure 5.3: Mean loss of cross-validation using a strain sensor (S1P4).



(a) Model1.



(b) Model3.

Figure 5.4: Mean loss of cross-validation using an accelerometer (A3ZP8).

5.3 Experimental Results for Bridge C

We implemented the three proposed networks and four tasks listed in Fig. 5.2, Table 5.1. The networks were written in Python 2.7 and implemented on Chainer [68] 5.0.0. They were accelerated by an NVIDIA Tesla P100 GPU using CUDA [60] 8.

We employed the AMSGrad [72] as an alternative version of the Adam [37] optimizer for stochastic gradient descent whose parameters were set as defined in the Chainer implementation by default. The mini-batch size was set at 10.

The CNN models were evaluated mainly in terms of the final generalization performance after loss convergence and before overfitting. The performance was visualized utilizing statistical metrics. For Task 1, we used receiver operating characteristic (ROC) curves and the area under the curves (AUCs). For Tasks 2 and 3, we used histograms, and the model accuracy (ACC) was evaluated in terms of mean absolute errors (MAEs). For Task 4, we made confusion matrices to evaluate precision and recall.

It should be noted that the implementation was improved in two aspects in relation to previous versions in previous papers [34, 35], so that the models could achieve superior performance. First, we corrected the mechanism of early stopping [98], which had not worked correctly in the previous versions. Second, we modified the algorithm of random shift of vehicle timestamps for Bridge C so the strain responses would not fade out of the input window.

5.3.1 Loss Convergence

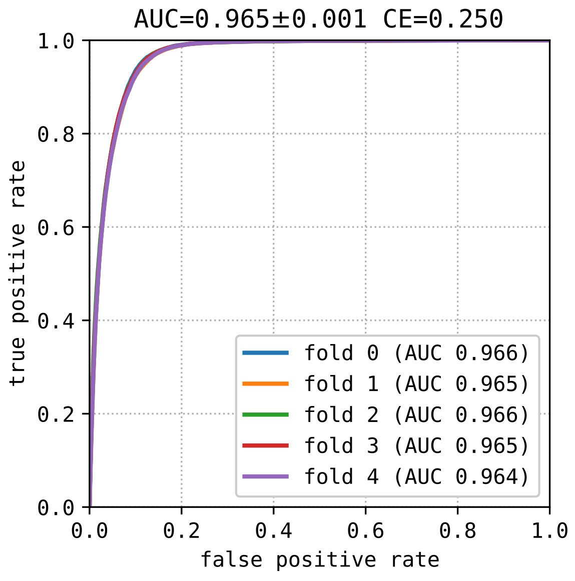
The networks were trained over 200 epochs. That is, each sample was fed to the networks 200 times. The fivefold cross-validation for Model 3 required about 11 days for the 30-day strain data and 4 days for the 163-day vibration data. Fig. 5.3 shows the time evolution of the training and validation losses for Models 1 and 3 using strain sensor S1P4. Fig. 5.4 shows the time evolution of losses for accelerometer A3ZP8. The models converge in the first 50 epochs and start overfitting where the training losses become lower than the validation losses. As described in Section 5.2, the evaluation processes were performed with models whose validation losses were minimum ones, in the same fashion as the early stopping approach [98].

5.3.2 Detectability with Strain Sensors

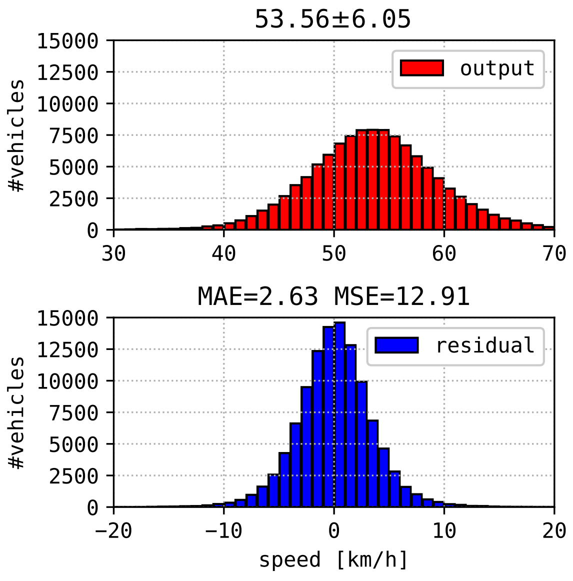
Fig. 5.5 shows the LtoR vehicle detection accuracy for Model 3 using strain sensor S1P4. Similarly, Fig. 5.6 shows the RtoL vehicle detection accuracy using strain sensor S4P4. As shown in Fig. 4.1 (a), the sensors were installed to be under the wheels of LtoR or RtoL vehicles.

For vehicle detection Task 1, Fig. 5.5 (a) shows the ROC curves for the fivefold evaluation datasets. According to the AUC, the model achieved high accuracy. The standard deviation of the AUCs was near zero, meaning that the model achieved stable accuracy levels. Fig. 5.6 (a) shows the same tendency.

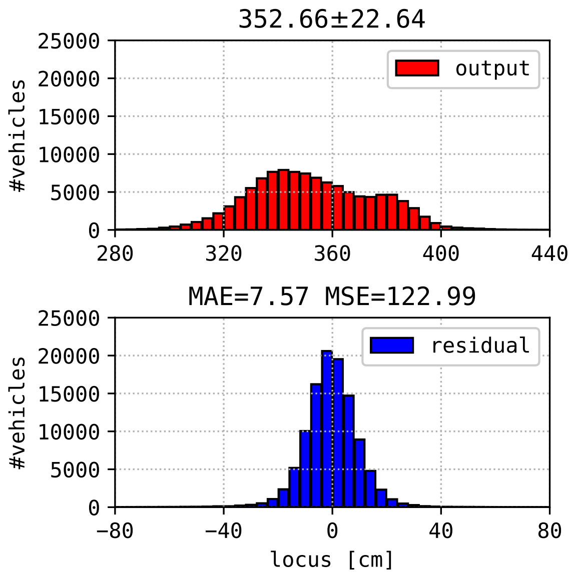
For speed estimation Task 2, Fig. 5.5 (b) and Fig. 5.6 (b) show the total evaluation results for



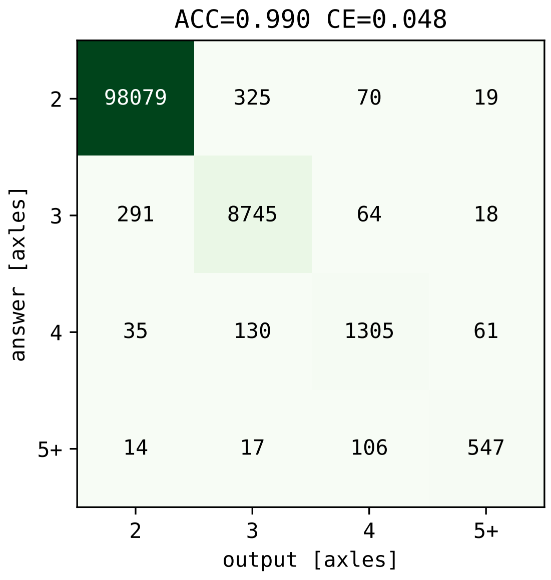
(a) Task 1.



(b) Task 2.



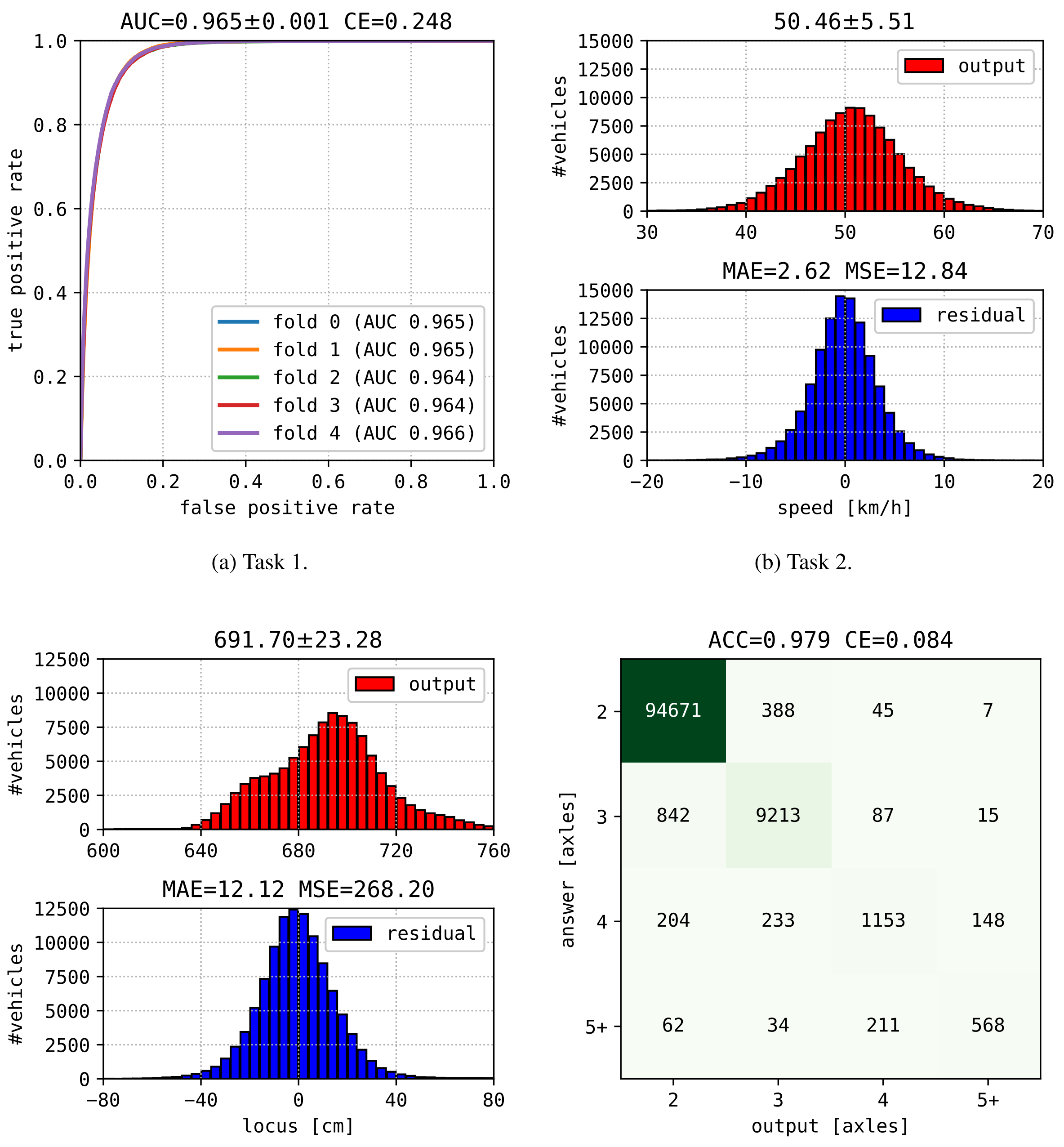
(c) Task 3.



(d) Task 4.

Figure 5.5: Model 3 performance of LtoR vehicle detection (S1P4).

the cross-validation. The upper graphs show the distribution of the estimated traveling speeds, and the lower graphs show the distribution of the residuals. The residuals were distributed around 0 km/h following a unimodal distribution. For LtoR vehicles, the MAE was 2.63 km/h, while the standard deviation of prediction (SDP) was 6.05 km/h. For RtoL ones, the MAE was 2.62 km/h and the SDP was 5.51 km/h. Consequently, a strain signal for 8 s sampled at 200 Hz contains information correlated with the speed, and Model 3 succeeded in extracting the corresponding features. The validity of the speed estimates is explored later in Section 5.3.4. It



(a) Task 1. (b) Task 2.

(c) Task 3. (d) Task 4.

Figure 5.6: Model 3 performance of RtoL vehicle detection (S4P4).

should be noted that the upper graph is slightly sharper than the ground truth. For LtoR vehicles, the standard deviation of the estimation result was 6.05 km/h, whereas that of the ground truth was 7 km/h. Consequently, the estimation results were biased toward the average speed, and the contribution of the outlier vehicles was weaker than the bias.

For locus estimation Task 3, Fig. 5.5 (c) and Fig. 5.6 (c) show the total evaluation results for the cross-validation. The upper graphs show the distribution of the estimated traveling positions, and the lower graphs show the distribution of the residuals. The histograms show the same tendencies as those for Task 2. The residuals were distributed around 0 cm. For LtoR vehicles, the MAE was 7.57 cm, while the SDP was 22.64 cm. For RtoL ones, the MAE was 12.12 cm and the SDP was 23.28 cm. Model 3 succeeded in extracting some features correlated with the loci. According to the MAE, Model 3 was highly sensitive to locus identification, with errors less than the wheel widths. The validity of the estimates is explored in Section 5.3.4.

Fig. 5.5 (d) and Fig. 5.6 (d) show the evaluation results for axle counting Task 4. The figures show the confusion matrices for 2-axle, 3-axle, 4-axle, and more-than-4-axle vehicles. Focusing on the sum of true-positive cells on the diagonal line, Model 3 achieved moderate values for precision and recall.

5.3.3 Detectability with Acceleration Sensors

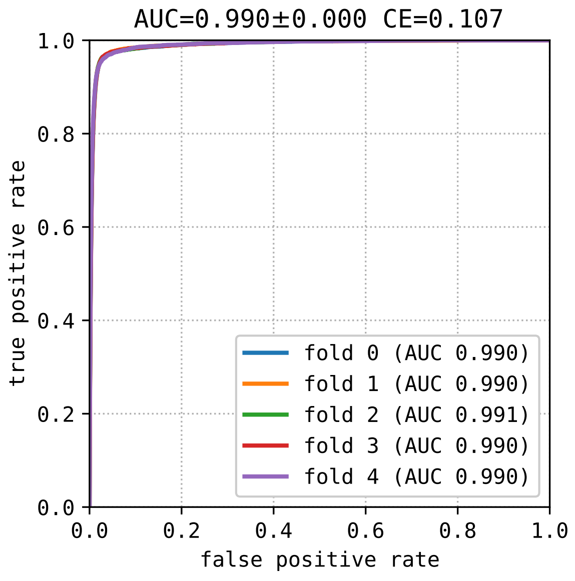
Fig. 5.7 and Fig. 5.8 show the evaluation results for vehicle detection using accelerometer A3ZP8 and Model 3 on Bridge C. Fig. 5.7 shows the results for LtoR vehicle detection, while Fig. 5.8 shows the results for RtoL vehicle detection. As shown in Fig. 4.1 (a), A3ZP8 was installed at the center of the bridge span. It should be noted that 2-axle vehicles were ignored, as described in Section 5.2, because the light weight of passenger cars made detecting them too difficult. Instead, we increased the dataset volume to cover 163 days.

Fig. 5.7 (a) and Fig. 5.8 (a), respectively, show the ROC curves for LtoR and RtoL vehicle detection. According to the AUCs, the model achieved vehicle (lane) detectability with high and stable accuracy.

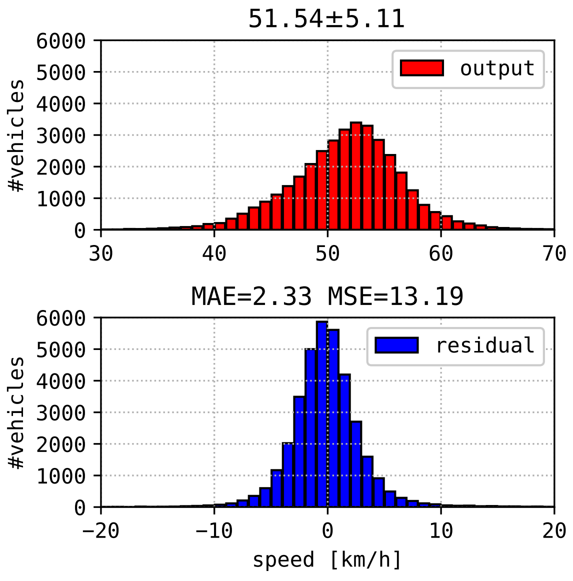
For speed estimation Task 2, Fig. 5.7 (b) and Fig. 5.8 (b) show the total evaluation results for the cross-validation. The upper graphs show the distribution of estimated speeds, and the lower graphs show the distribution of residuals. The MAEs were 2.33 km/h and 2.18 km/h, while the SDPs were 5.11 km/h and 5.24 km/h. As a result, a vibration sequence for 8 s sampled at 200 Hz contains some information correlated with the speed, as we found for the strain-based results.

For locus estimation Task 3, Fig. 5.7 (c) and Fig. 5.8 (c) show the evaluation results. The upper graphs show the distribution of estimated loci, and the lower graphs show the distribution of residuals. The MAEs were 11.30 cm and 11.06 cm, while the SDPs were 13.25 cm and 16.16 cm. Compared with the locus estimation using the strain sensors shown in Fig. 5.5 (c) and Fig. 5.6 (c), the MAEs were not much smaller than the standard deviations. We therefore have some doubts about the validity of locus prediction, and this is explored in Section 5.3.4.

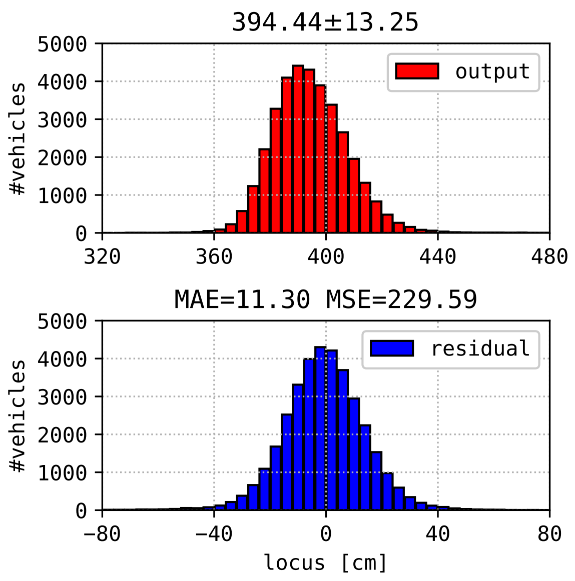
Fig. 5.7 (d) and Fig. 5.8 (d) show the confusion matrices for axle counting Task 4. The accuracy was much lower than the axle counting consulting strain sensors shown in Fig. 5.5 (d) and Fig. 5.6 (d). Focusing on vehicles with more than 3 axles, both precision and recall were quite low, indicating that the model is impractical.



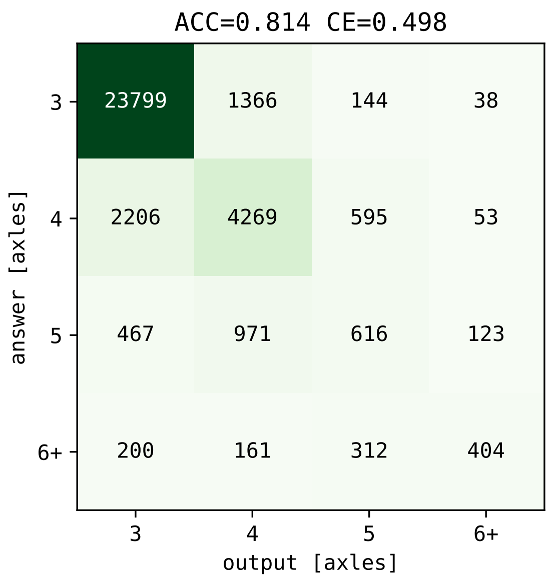
(a) Task 1.



(b) Task 2.



(c) Task 3.

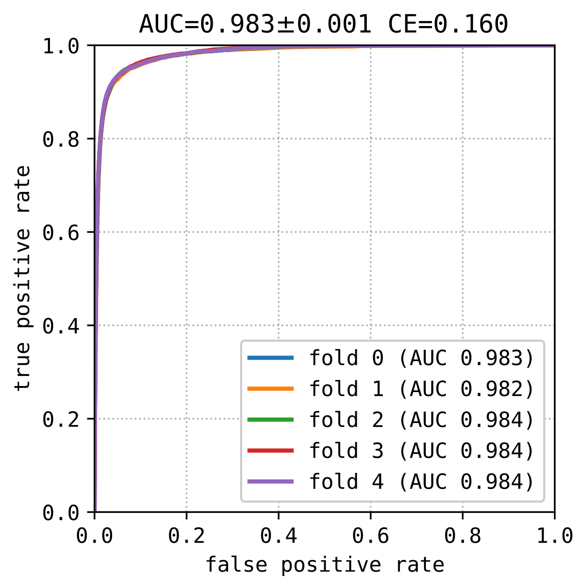


(d) Task 4.

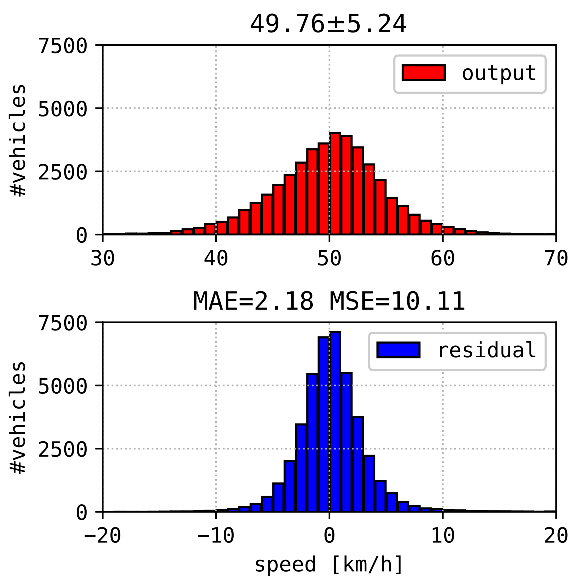
Figure 5.7: Model 3 performance of LtoR vehicle detection (A3ZP8).

5.3.4 Validation of Detectability

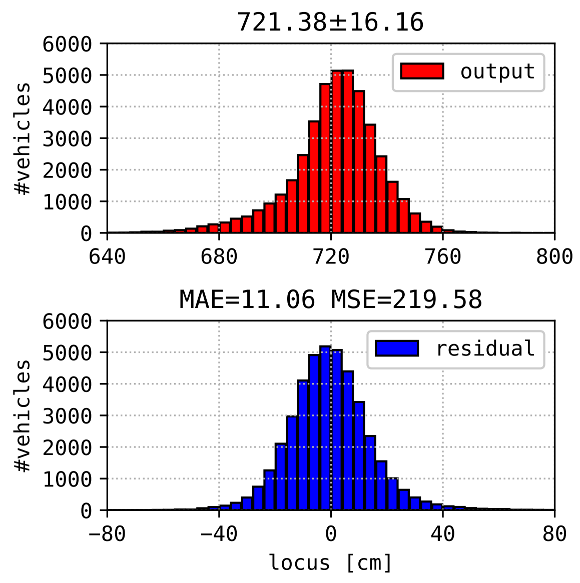
As noted, we had some doubts about the validity of locus estimation, particularly when using the accelerometer A3ZP8. To clarify the issue, we visualized the correlation between the ground-truth g and predicted values y for Tasks 2 and 3. Each graph indicates the density of the (g, y) distribution obtained by Gaussian kernel density estimation.



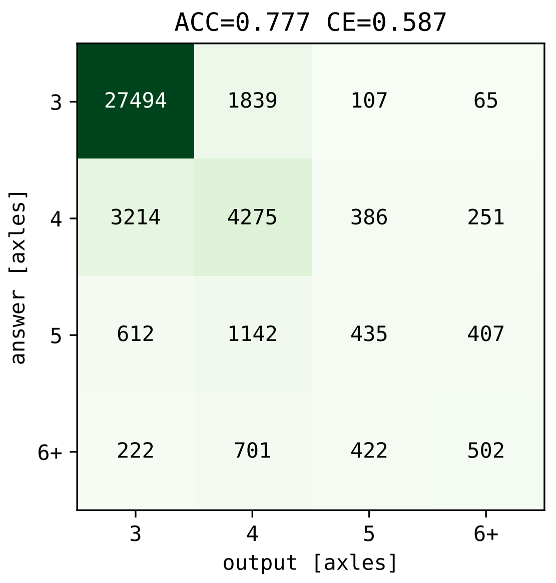
(a) Task 1.



(b) Task 2.



(c) Task 3.

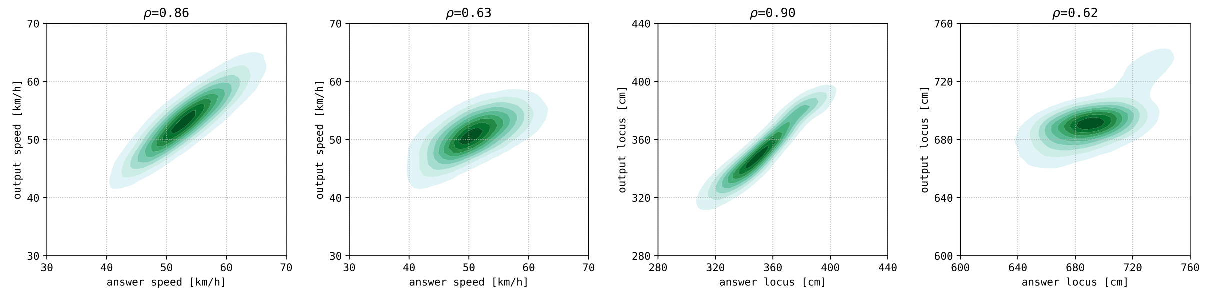


(d) Task 4.

Figure 5.8: Model 3 performance of RtoL vehicle detection (A3ZP8).

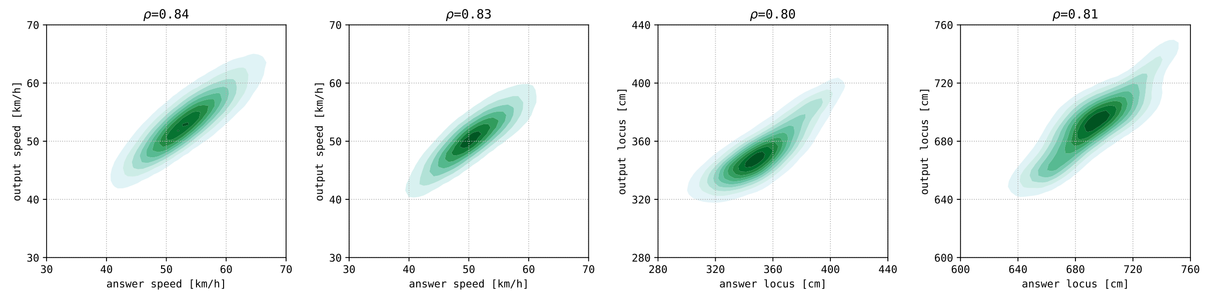
Fig. 5.9 and Fig. 5.10 indicate the validity of vehicle detection from strain sensors S1P4 and S4P4, respectively. These graphs show the correlation graphs for speed estimation Task 2 and locus estimation Task 3. For both tasks, the (g, y) clusters are clearly spread along the $g = y$ line. The correlation coefficients ρ were more than 0.8 for vehicles passing in their lanes. Thus, the models certainly captured some features correlated with the speeds and loci. It should be noted that the ground truth was obtained through video analysis via the TSS.

Fig. 5.11 indicates the validity of vehicle detection from the accelerometer. Fig. 5.11 (a)



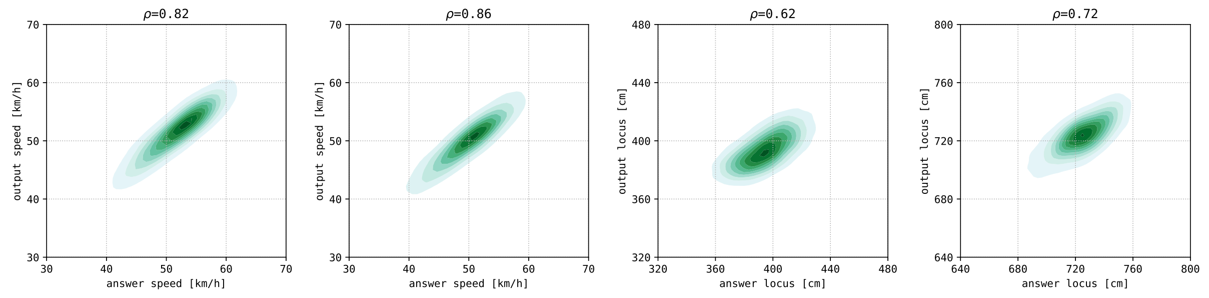
(a) Task 2 LtoR. (b) Task 2 RtoL. (c) Task 3 LtoR. (d) Task 3 RtoL.

Figure 5.9: Validation of vehicle detection performance by Model 3 (S1P4).



(a) Task 2 LtoR. (b) Task 2 RtoL. (c) Task 3 LtoR. (d) Task 3 RtoL.

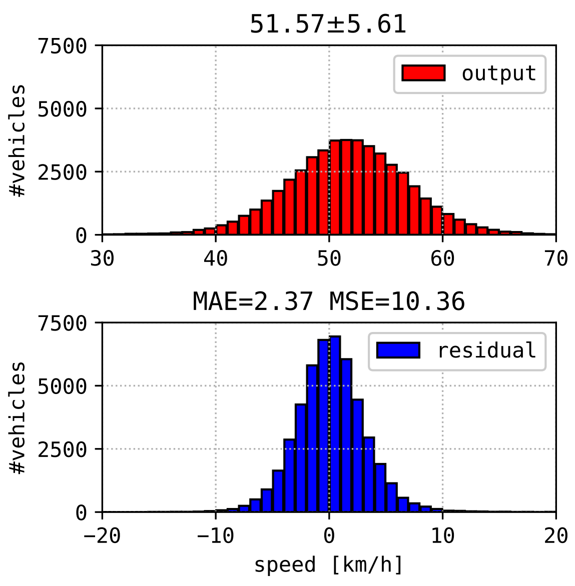
Figure 5.10: Validation of vehicle detection performance by Model 3 (S4P4).



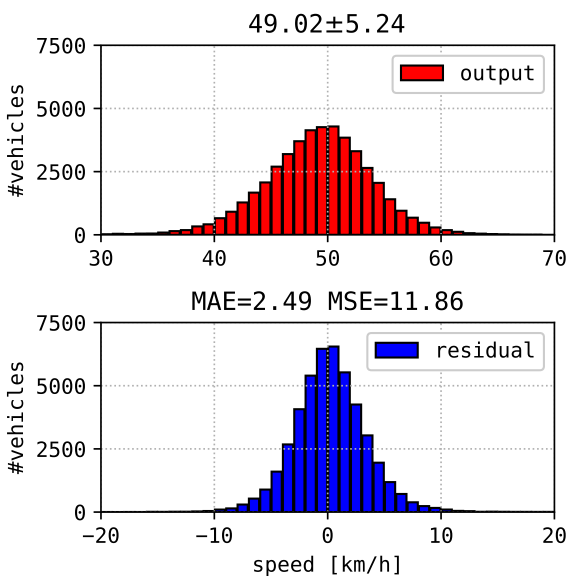
(a) Task 2 LtoR. (b) Task 2 RtoL. (c) Task 3 LtoR. (d) Task 3 RtoL.

Figure 5.11: Validation of vehicle detection performance by Model 3 (A3ZP8).

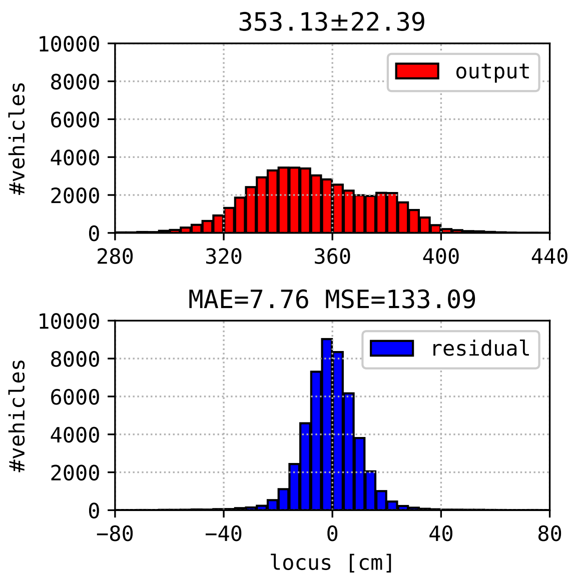
and Fig. 5.11 (b) show the correlation graphs for speed estimation Task 2. Fig. 5.11 (c) and Fig. 5.11 (d) show the correlation graphs for locus estimation Task 3. For speed estimation Task 2, the (g, y) clusters are clearly spread along the $g = y$ line. The correlation coefficients ρ are more than 0.8, indicating that the models can estimate the speeds. For locus estimation Task 3, the (g, y) clusters presented a circular pattern rather than a linear pattern. The correlation coefficients were not high but were obviously positive, and we believe that the models captured some features correlated with loci. The weak correlation may be because of the following two reasons. Large vehicles with more than two axles are generally wide and may not move freely



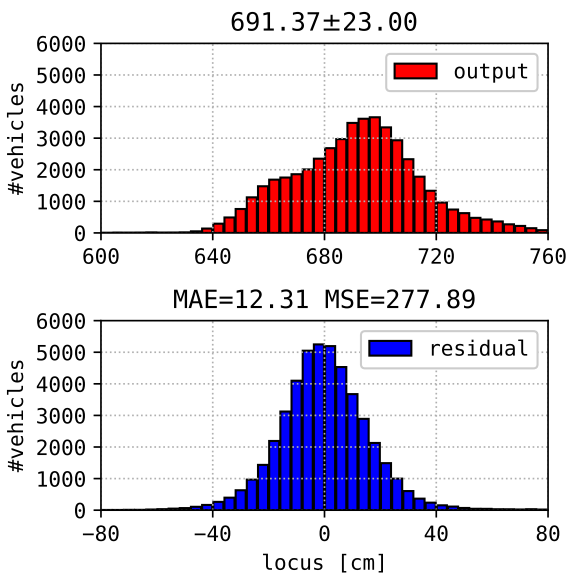
(a) Task 2 LtoR (S1P4).



(b) Task 2 RtoL (S4P4).



(c) Task 3 LtoR (S1P4).



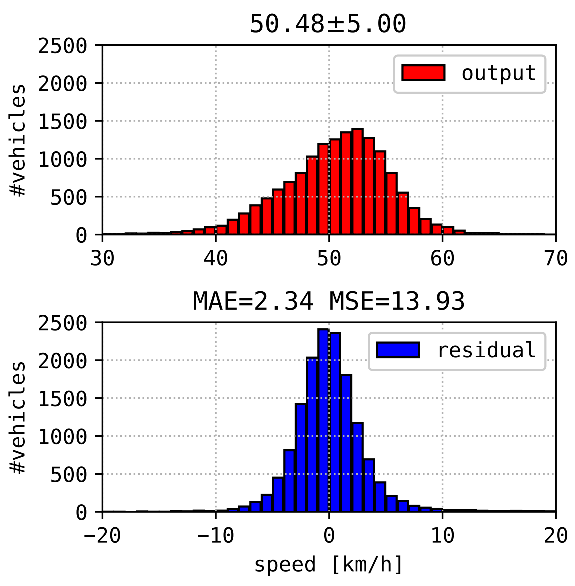
(d) Task 3 RtoL (S4P4).

Figure 5.12: Performance of Model 3 using strain sensors under heavy traffic.

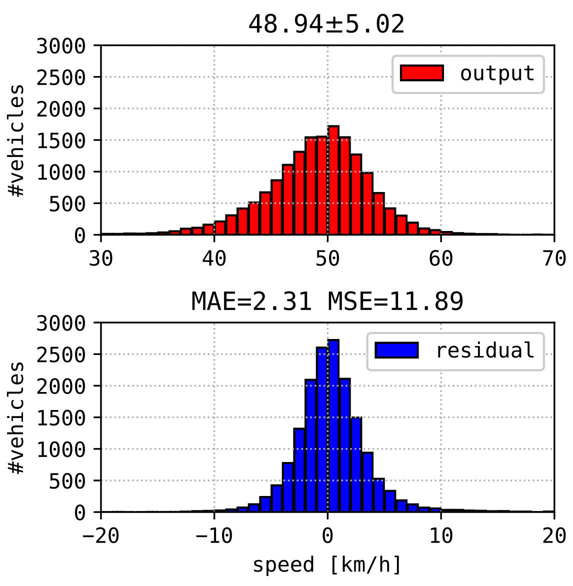
within the lane. Moreover, A3ZP8 was installed at the center of the bridge span, which was more than 30 m from the camera.

5.3.5 Accuracy under Heavy Traffic

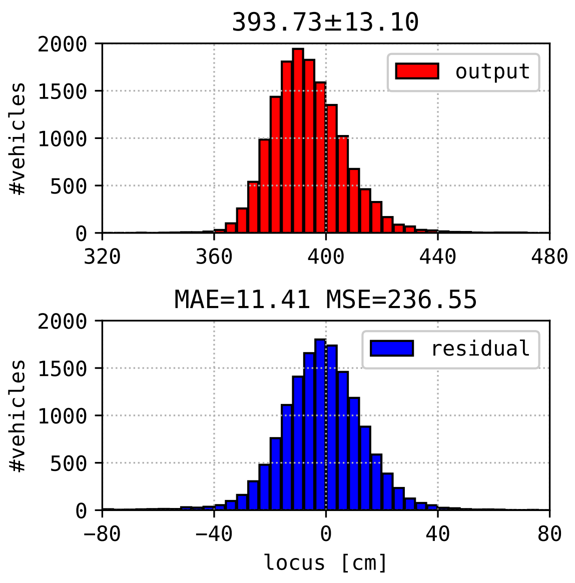
We had some doubts that the detection accuracy using our proposed networks might be affected by the traffic volume at various times. As shown in Fig. 4.4 (d), the interval between a



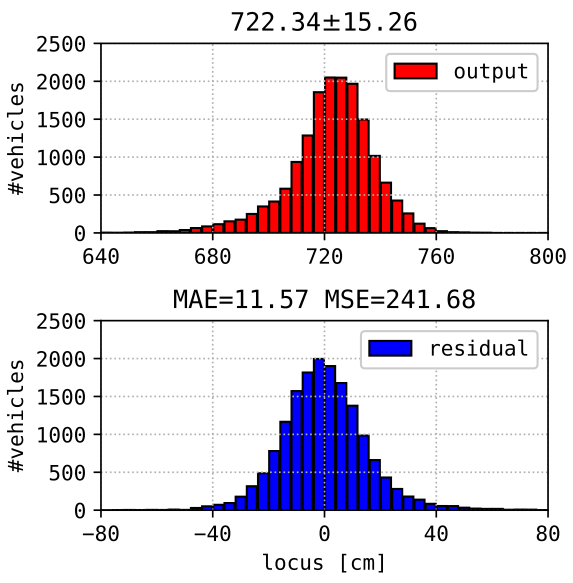
(a) Task 2 LtoR (A3ZP8).



(b) Task 2 RtoL (A3ZP8).



(c) Task 3 LtoR (A3ZP8).



(d) Task 3 RtoL (A3ZP8).

Figure 5.13: Performance of Model 3 using an accelerometer under heavy traffic.

vehicle disappearing and the next one appearing in the camera field could vary considerably. To investigate the effect of traffic volume, we calculated the detection performance if we ignored vehicles for which the gap to the following vehicle exceeded 2 s. Fig. 5.12 and Fig. 5.13 show the histograms of these modified estimation results and the residuals for the strain data and vibration data, respectively. Compared with the results in Section 5.3.2 and Section 5.3.3, we can confirm that any degradation in accuracy was insignificant.

Table 5.2: Comparison of models for LtoR vehicles (S1P4).

Task	Index	Model 1	Model 2	Model 3	Unit
1	CE	0.26	0.26	0.25	-
2	MAE	2.73	2.69	2.63	km/h
	SDP	5.74	6.10	6.05	km/h
3	MAE	7.70	7.82	7.57	cm
	SDP	22.01	22.41	22.64	cm
4	CE	0.06	0.05	0.05	-

5.3.6 Comparison of Models

As shown in Fig. 5.2, we designed three derivative CNN architectures. Each row in Table 5.2 indicates the accuracy of a task learned by the three models individually. Overall, the deepest model, Model 3, achieved superior accuracy to the other two models. On the other hand, Model 2 failed to improve on the shallowest model, Model 1.

For vehicle detection Task 1, the overall cross entropy (CE) across five folds was slightly decreased from 0.26 to 0.25 by inserting some residual blocks. For speed estimation Task 2, Model 3 achieved superior performance to the other models, but there was little superiority in terms of MAE. However, the shape of the speed distribution estimated by Model 3 was much closer to the ground truth than that for Models 1 and 2, which can be confirmed in terms of the SDP. The SDP achieved by Model 3 was 6.05 km/h, whereas the standard deviation of ground truth was 7.01 km/h. Models 1 and 2 had poor sensitivity toward detecting vehicles that were either very slow or very fast. For locus estimation Task 3, Model 3 achieved superior performance to the other models, as for Task 2. The SDP of Model 3 was 22.64 cm, whereas the standard deviation of ground truth was 25.04 cm. In terms of the SDP, Model 1, the simplest model, had poor sensitivity toward detecting vehicles that ran too far to the left or too far to the right in the lane. Model 2 was attracted by outlier loci but failed to improve its MAE. For axle counting Task 4, Model 3 achieved slightly superior performance to the other models.

5.3.7 Comparison of Sensor Positions

As illustrated in Fig. 4.1 (a), we installed four strain sensors on Bridge C, denoted S1P4, S2P4, S3P4, and S4P4, from left to right in Fig. 4.1 (a). These sensors were located in a line orthogonal to the bridge axis. For most LtoR vehicles, sensor S1P4 was the closest to the

Table 5.3: Comparison of sensor positions for Model 3.

		LtoR				RtoL				
Task	Loss	S1P4	S2P4	S3P4	S4P4	S1P4	S2P4	S3P4	S4P4	Unit
1	CE	0.25	0.34	0.24	0.23	0.38	0.29	0.24	0.25	-
2	MAE	2.63	3.11	2.88	2.81	3.70	3.19	2.60	2.62	km/h
3	MAE	7.57	8.92	10.87	11.46	16.87	15.06	11.90	12.12	cm
4	CE	0.05	0.06	0.05	0.05	0.11	0.09	0.08	0.08	-

Table 5.4: Comparison of accelerometer axes for Model 3.

		LtoR			RtoL			
Task	Loss	A3XP8	A3YP8	A3ZP8	A3XP8	A3YP8	A3ZP8	Unit
1	CE	0.26	0.23	0.11	0.25	0.21	0.16	-
2	MAE	3.10	3.21	2.33	2.70	2.62	2.18	km/h
3	MAE	13.00	12.68	11.30	14.92	13.06	11.06	cm
4	CE	0.73	0.70	0.50	0.74	0.67	0.59	-

wheels, while sensor S4P4 was the closest to most RtoL vehicle axles. Each row of Table 5.3 shows the detection accuracy of LtoR and RtoL vehicles using the four strain sensors. In general, the sensors S1P4 and S4P4 achieved superior accuracy to the other three sensors for LtoR and RtoL vehicles, respectively.

5.3.8 Comparison of Acceleration Directions

As described in Section 4.1, we installed an accelerometer with three axes, and the vertical axis (Z) was denoted A3ZP8. The sensor has other two horizontal axes, X (bridge axis) and Y (orthogonal axis). Each row of Table 5.4 shows the detection accuracy of LtoR and RtoL vehicles using the three acceleration axes. In general, the A3ZP8 achieved superior accuracy to the other two axes, and A3XP8 achieved quite low performance. It seems that the Z-axis responded greatly to the passing vehicles, and the remaining two axes vibrated to a certain degree, but their amplitudes were much smaller than the vertical axis.

Table 5.5: Effect of LPF on LtoR vehicle detectability by Model 3.

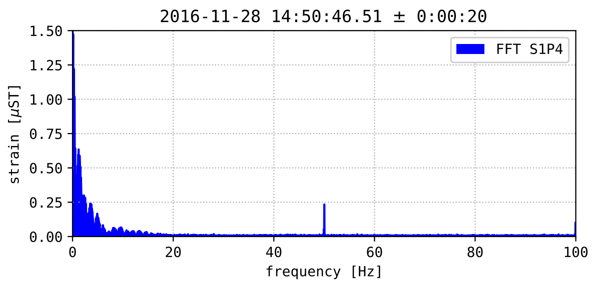
		S1P4				A3ZP8				
Task	Loss	1 Hz	2 Hz	6 Hz	Raw	10 Hz	20 Hz	30 Hz	Raw	Unit
1	CE	0.26	0.26	0.26	0.25	0.19	0.13	0.13	0.11	-
2	MAE	3.35	3.18	2.90	2.63	3.14	2.65	2.47	2.33	km/h
3	MAE	11.11	10.70	8.31	7.57	13.49	12.32	12.09	11.30	cm
4	CE	0.08	0.07	0.05	0.05	0.67	0.60	0.57	0.50	-

5.3.9 Effect of the Low-Pass Filter (LPF)

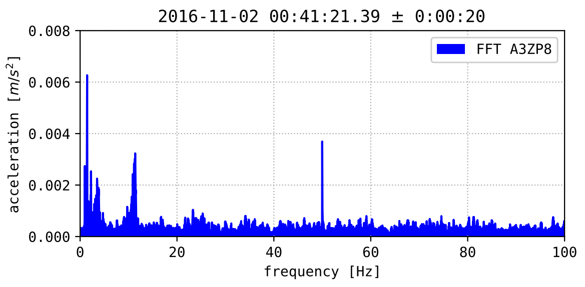
It is preferable to use the lowest feasible sampling rate from the perspective of power consumption and the installation cost of the sensing environment. In addition, we must identify the frequency band that best correlates with vehicle detectability in the four prediction tasks in order to elucidate the mechanism by which the deep sensing models identify individual vehicles. Therefore, we applied an LPF to the strain and vibration data before their input to Model 3 and compared the resulting performance with the results for models that were fed original signals. The cutoff frequencies for the LPF were determined by considering the dominant frequencies in the power spectra shown in Fig. 5.14. The isolated peaks at 50 Hz were caused by power supply noise. Table 5.5 lists the comparison results for the four tasks. For the strain data, components over 6 Hz have little influence on vehicle detectability. For the vibration data, components over 30 Hz have little influence on vehicle detectability.

5.3.10 Effect of Training Data Volume

For the experiments in Section 5.3, we collected data from vehicles that crossed Bridge C over a six-month period. It is preferable to use the smallest feasible dataset from the perspective of reducing the training cost at system initialization. We thus conducted an additional experiment to investigate the contribution of dataset volume to detection performance. We subsampled each *trainval* dataset for fivefold cross-validation so the volume of sampled *trainval* data was varied from 10% to 100%. Fig. 5.15 shows the learning curves for the validation scores. For regressions in Tasks 2 and 3, the dataset volume barely influences the accuracy, and we may even decrease the number of samples to 10% (19.2 hours) in extreme cases. In contrast, we must use more than about 50% (96 hours) of the original *trainval* data for Task 4. The volume

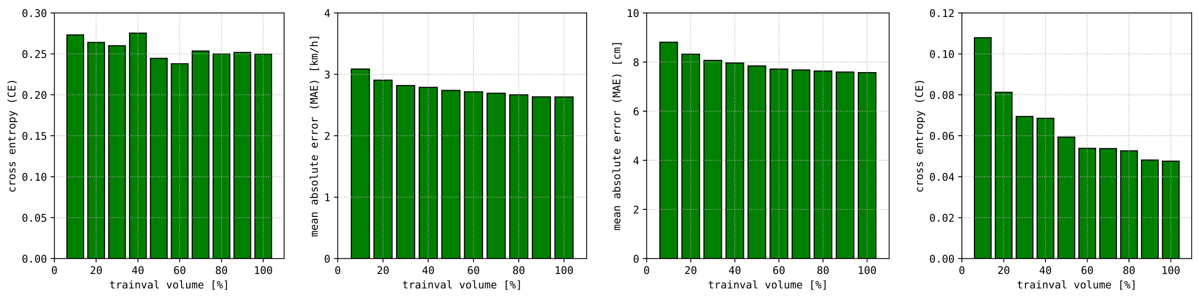


(a) Strain sensor (S1P4).



(b) Accelerometer (A3ZP8).

Figure 5.14: Power spectra of sensor data on Bridge C.



(a) Task 1. (b) Task 2. (c) Task 3. (d) Task 4.

Figure 5.15: Effect of training data volume on LtoR vehicle detection by Model 3 (S1P4).

greatly influences the accuracy for axle counting Task 4 because of the rarity of multi-axle vehicles in the *trainval* data.

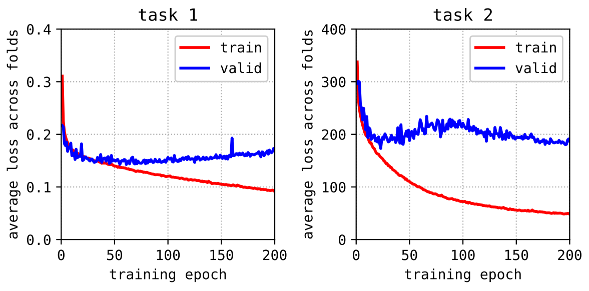
5.4 Experimental Results for Bridge S

We also evaluated Model 3 for vehicles in the first (left) lane on Bridge S. As described in Section 4.1, we used three types of strain sensors on Bridge S. First, strain sensors were used on vertical stiffeners, denoted VSG2B1, VSG2B2, VSG4B1, and VSG4B2. Second, strain sensors were used on lower flanges, denoted LSG2C and LSG4C. Third, strain sensors were used on bridge supports, denoted RG2A1, RG2A2, RG4A1, and RG4A2.

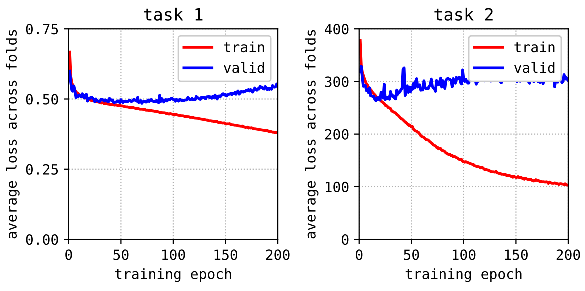
We implemented Model 3 and Tasks 1 and 2, listed in Fig. 5.2, Table 5.1. The network was written in Python 2.7 and implemented on Chainer [68] 5.0.0. The network was accelerated by an NVIDIA Tesla V100 GPU, using CUDA [60] 9.2.

We employed the AMSGrad [72] as an implementation of stochastic gradient descent whose parameters were set as defined in the Chainer implementation by default. The mini-batch size was set at 10.

It should be noted that a 20% dropout was inserted between the two convolutional layers in

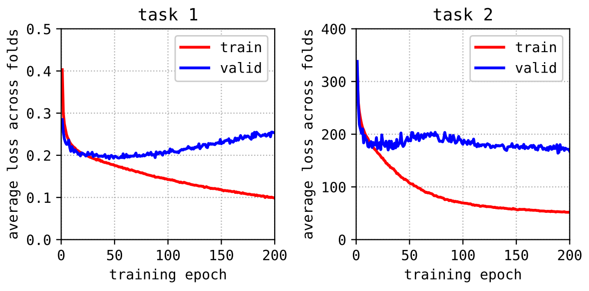


(a) VSG2B1.

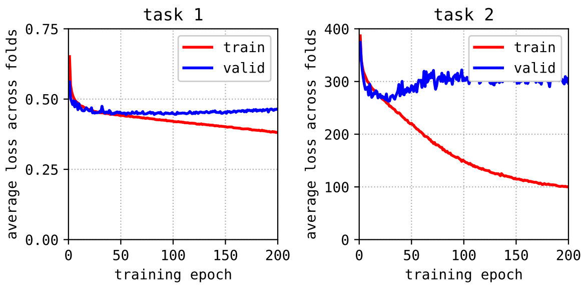


(b) VSG4B1.

Figure 5.16: Mean loss of cross-validation using sensors on vertical stiffeners.

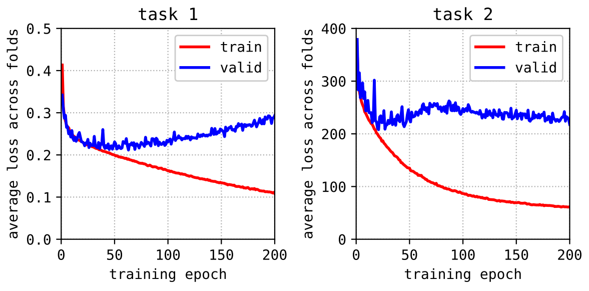


(a) LSG2C.

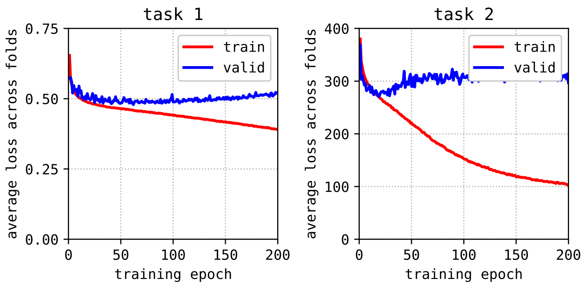


(b) LSG4C.

Figure 5.17: Mean loss of cross-validation using sensors on lower flanges.



(a) RG2A1.



(b) RG4A1.

Figure 5.18: Mean loss of cross-validation using sensors on girder supports.

each residual block.

5.4.1 Loss Convergence

The networks were trained over 200 epochs, and each sample was fed to the networks 200 times. The fivefold cross-validation for Model 3 required about 3 days. Fig. 5.16 shows the time evolution of the training and validation losses for strain sensors on vertical stiffeners. Fig. 5.17 shows the time evolution of losses for strain sensors on lower flanges. Fig. 5.18 shows the time evolution of losses for strain sensors on girder supports. The models converge in the first 50 epochs and start overfitting where the training losses become lower than the validation losses. As described in Section 5.2, the evaluation processes were performed with models whose validation losses were minimum in the same fashion as the early stopping approach [98].

5.4.2 Detectability with Strain Sensors on Vertical Stiffeners

In this section, the performance for vehicle detection using strain sensors on vertical stiffeners is evaluated. The graphs for Tasks 1 and 2 show the same tendencies as the results for Bridge C.

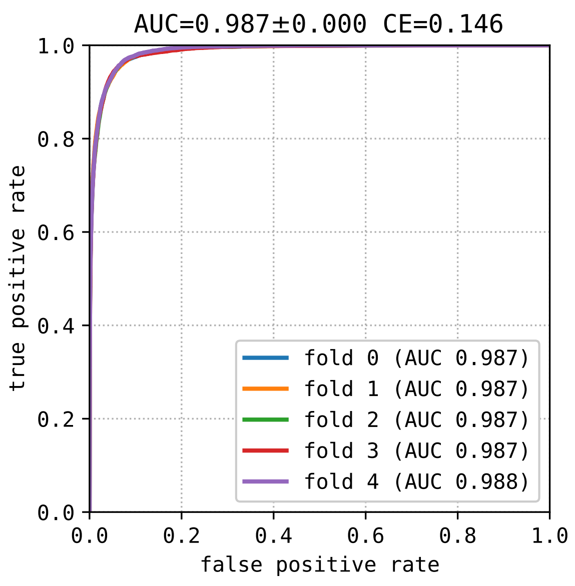
For vehicle detection Task 1, Fig. 5.19 (a) and Fig. 5.20 (a) show the individual ROC curves for fivefold cross-validation using VSG2B1 and VSG4B1, respectively. According to the AUCs, Model 3 using VSG2B1 detected vehicles accurately in the first lane, which is a good demonstration of lane detectability. It is notable that Model 3 using VSG2B1 achieved superior accuracy to Model 3 using VSG4B1. This may be explainable by the sensor positions. VSG2B1 was located near the wheels in the first lane, while VSG4B1 was located under the other lane. The standard deviation of the AUCs was near zero, meaning the model achieved stable accuracy levels.

For speed estimation Task 2, Fig. 5.19 (b) and Fig. 5.20 (b) show the total results for the cross-validation using VSG2B1 and VSG4B1, respectively. The upper graphs show the distribution of the estimated traveling speeds, and the lower graphs show the distribution of the residuals. The residuals were distributed around approximately 0 km/h, following a unimodal distribution. For VSG2B1, the MAE was 7.71 km/h, while the SDP was 16.55 km/h. For VSG4B1, the MAE was 10.65 km/h and the SDP was 12.60 km/h. Consequently, a strain signal for 4 s sampled at 100 Hz contains information correlated with the speed, and Model 3 succeeded in extracting the corresponding features. However, the MAEs may be inadequate for practical use.

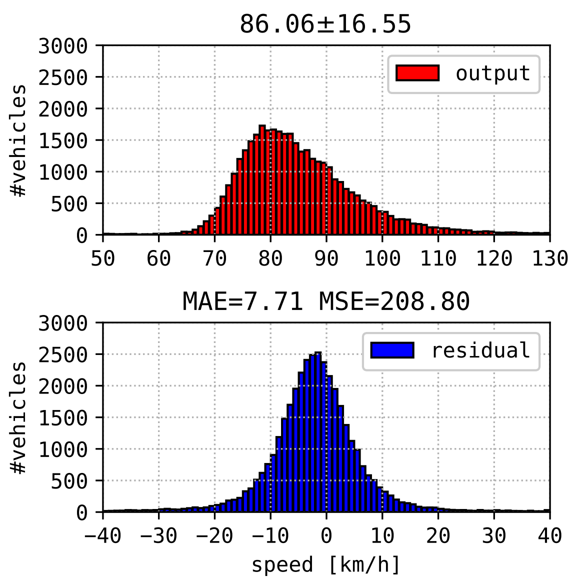
5.4.3 Detectability with Strain Sensors on Lower Flanges

In this section, the performance for vehicle detection using strain sensors on lower flanges is evaluated. The graphs for Tasks 1 and 2 show the same tendencies as the results for Section 5.4.2.

For vehicle detection Task 1, Fig. 5.21 (a) and Fig. 5.22 (a) show the individual ROC curves for fivefold cross-validation using LSG2C and LSG4C, respectively. According to the AUCs, Model 3 using LSG2C achieved superior accuracy to Model 3 using LSG4C. This increased accuracy may be explainable by the sensor positions. LSG2C was located near the wheels in the first lane, while LSG4C was located under the other lane. The standard deviation of the

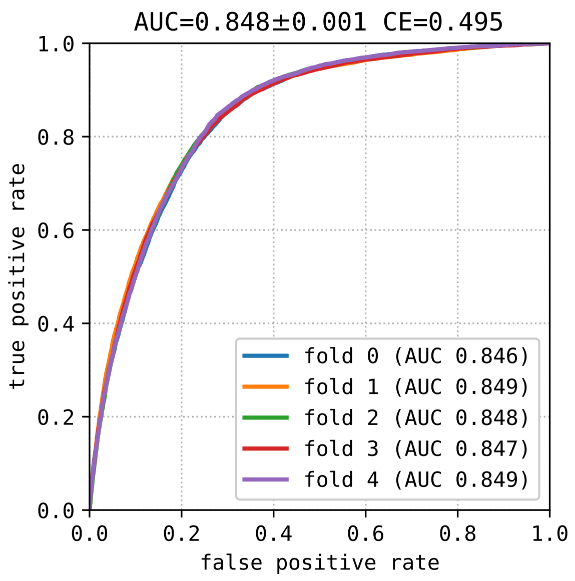


(a) Task 1.

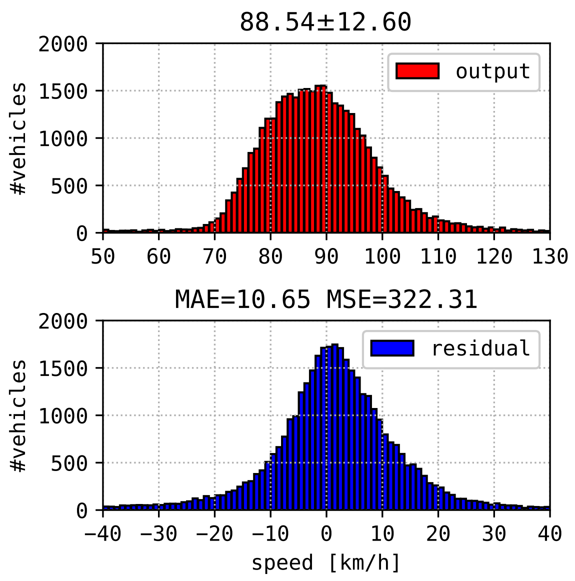


(b) Task 2.

Figure 5.19: Model 3 performance of first-lane vehicle detection (VSG2B1).



(a) Task 1.

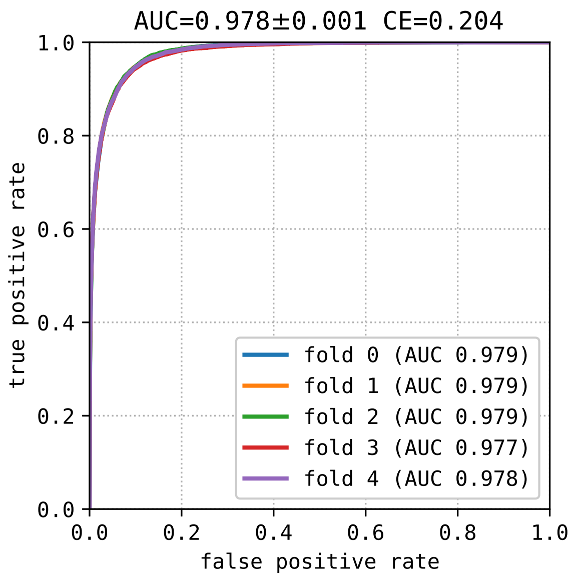


(b) Task 2.

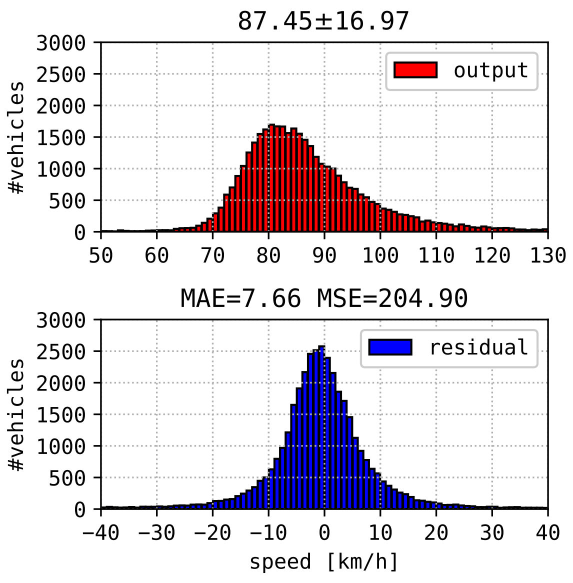
Figure 5.20: Model 3 performance of first-lane vehicle detection (VSG4B1).

AUCs was near zero, meaning both models achieved stable accuracy levels.

For speed estimation Task 2, Fig. 5.21 (b) and Fig. 5.22 (b) show the total results for the cross-validation using LSG2C and LSG4C, respectively. The upper graphs show the distribution of the estimated traveling speeds, and the lower graphs show the distribution of the residuals. The residuals were distributed around approximately 0 km/h, following a unimodal distribution.

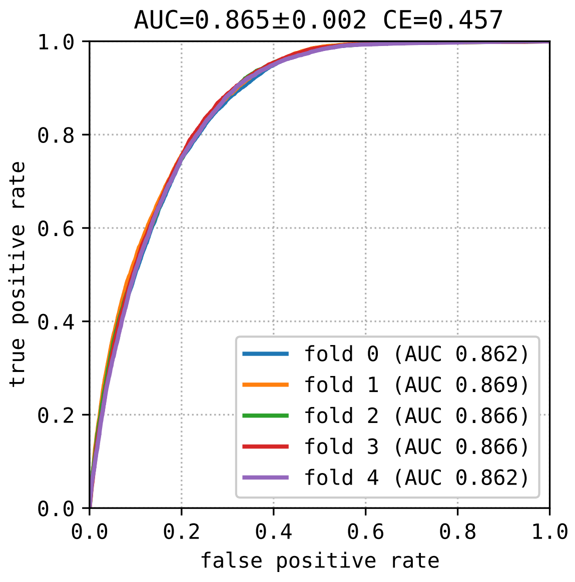


(a) Task 1.

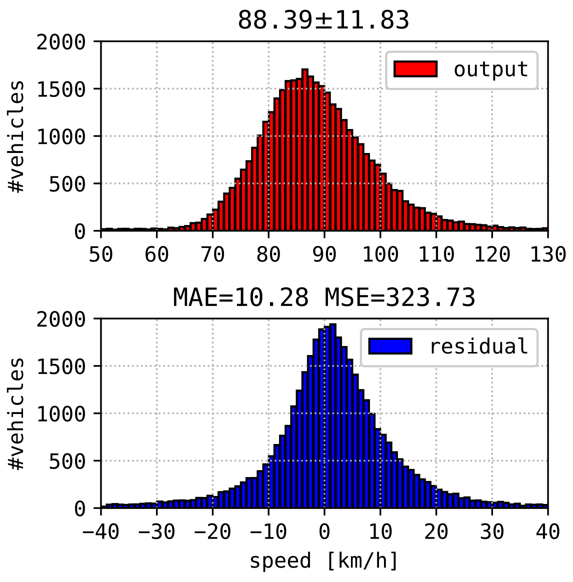


(b) Task 2.

Figure 5.21: Model 3 performance of first-lane vehicle detection (LSG2C).



(a) Task 1.



(b) Task 2.

Figure 5.22: Model 3 performance of first-lane vehicle detection (LSG4C).

For LSG2C, the MAE was 7.66 km/h, while the SDP was 16.97 km/h. For LSG4C, the MAE was 10.28 km/h and the SDP was 11.83 km/h. As for Task 1, Model 3 using LSG2C achieved superior accuracy to Model 3 using LSG4C.

5.4.4 Detectability with Strain Sensors on Girder Supports

In this section, performance for vehicle detection using strain sensors on girder supports at bridge entrance and exit are evaluated. The graphs for Tasks 1 and 2 show the same tendencies as the results for Section 5.4.2.

For vehicle detection Task 1, Fig. 5.23 (a) and Fig. 5.24 (a) show the individual ROC curves for fivefold cross-validation using RG2A1 and RG4A1, respectively. According to the AUCs, Model 3 using RG2A1 achieved superior accuracy to Model 3 using RG4A1. This increased accuracy may be explainable by the sensor positions. RG2A1 was located near the wheels in the first lane, while RG4A1 was located under the other lane. The standard deviation of the AUCs was near zero, meaning both models achieved stable accuracy levels.

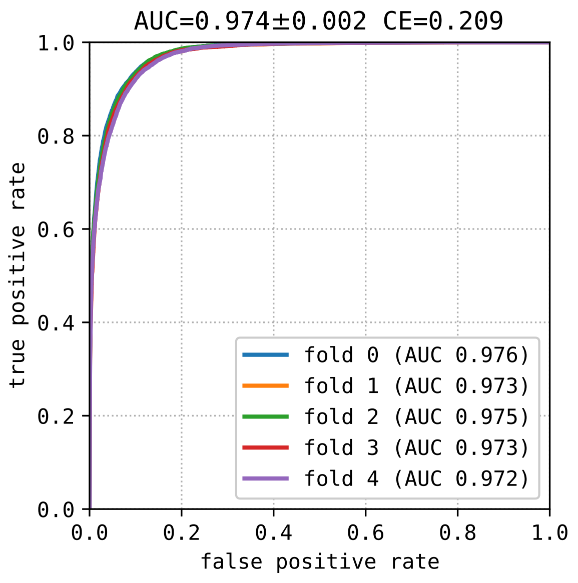
For speed estimation Task 2, Fig. 5.23 (b) and Fig. 5.24 (b) show the total results for the cross-validation using RG2A1 and RG4A1, respectively. The upper graphs show the distribution of the estimated traveling speeds, and the lower graphs show the distribution of the residuals. The residuals were distributed around approximately 0 km/h, following a unimodal distribution. For RG2A1, the MAE was 9.04 km/h, while the SDP was 15.09 km/h. For RG4A1, the MAE was 10.52 km/h and the SDP was 11.59 km/h. As for Task 1, Model 3 using RG2A1 achieved superior accuracy to Model 3 using RG4A1.

5.4.5 Validation of Detectability

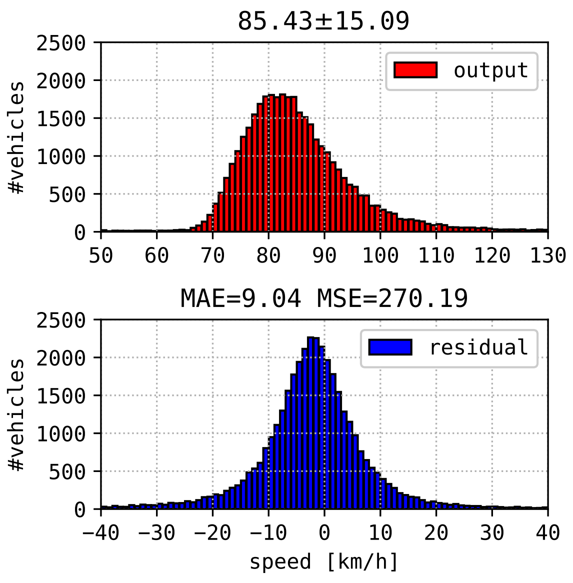
As for Bridge C, we had some doubts regarding the validity of speed estimation because the MAEs were not sufficiently lower than the standard deviation of the ground truth. To clarify the issue, we visualized the correlation between the ground-truth g and predicted values y for Task 2. Each graph indicates the density of the (g, y) distribution obtained by Gaussian kernel density estimation.

Fig. 5.25 indicates the validity of vehicle detection from strain sensors on vertical stiffeners. For sensors underneath girder G2, the (g, y) clusters are clearly spread along the $g = y$ line. The correlation coefficients ρ were about 0.7. Thus, the models certainly captured some features correlated with the speeds. For sensors underneath girder G4, the correlation coefficients were much lower than VSG2B1 and VSG2B2. This difference may be because of the closeness of the sensors to the first lane.

Fig. 5.26 indicates the validity of vehicle detection from strain sensors on lower flanges. For LSG2C, located underneath girder G2, the (g, y) clusters are clearly spread along the $g = y$ line. The correlation coefficient ρ was about 0.7. As a result, the models certainly captured some features correlated with the speeds. For LSG4C, located underneath girder G4, the correlation

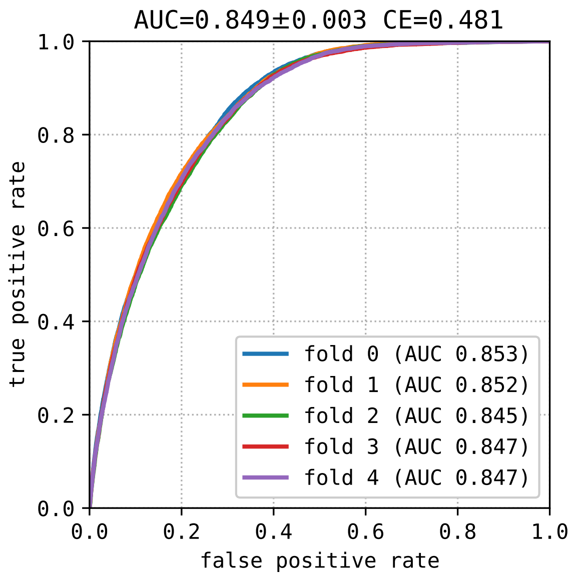


(a) Task 1.

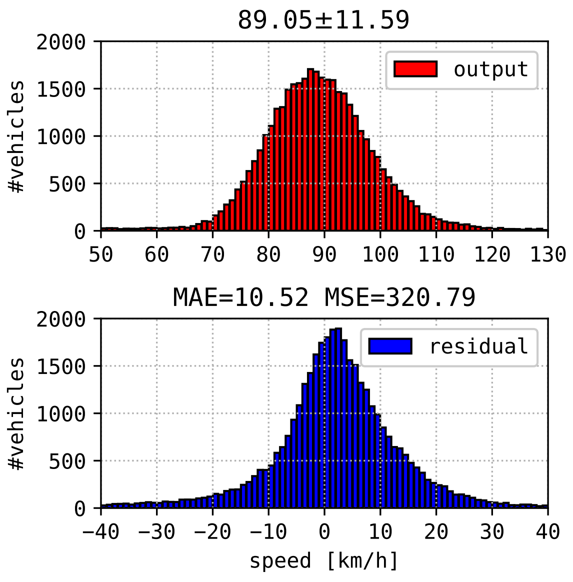


(b) Task 2.

Figure 5.23: Model 3 performance of first-lane vehicle detection (RG2A1).



(a) Task 1.

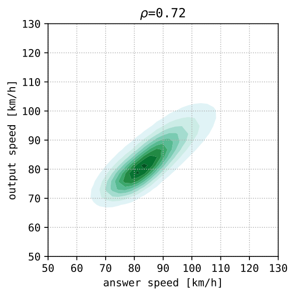


(b) Task 2.

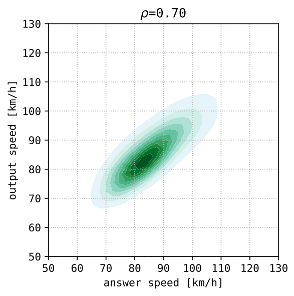
Figure 5.24: Model 3 performance of first-lane vehicle detection (RG4A1).

coefficient ρ was lower than that for VSG2B1 and VSG2B2. This may be explained by the same possible reason noted for VSG4B1 and VSG4B2.

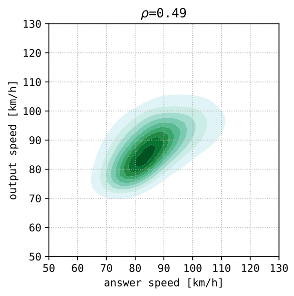
Fig. 5.27 indicates the validity of vehicle detection from strain sensors on girder supports at the bridge entrance and exit. According to the correlation coefficients ρ, the sensor RG2A1, which was located under the first lane at the bridge entrance, achieved the best detectability.



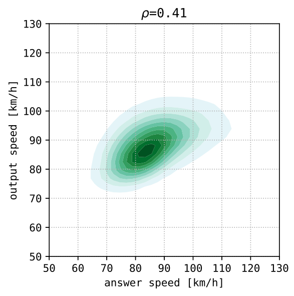
(a) VSG2B1.



(b) VSG2B2.

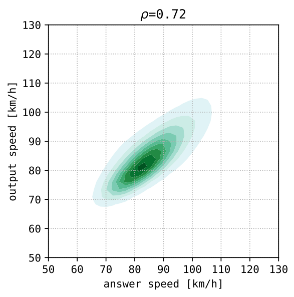


(c) VSG4B1.

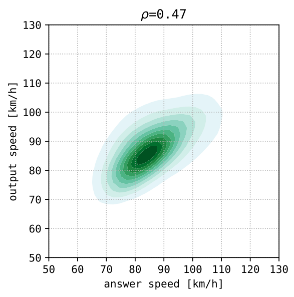


(d) VSG4B2.

Figure 5.25: Validation of vehicle detection using sensors on vertical stiffeners.

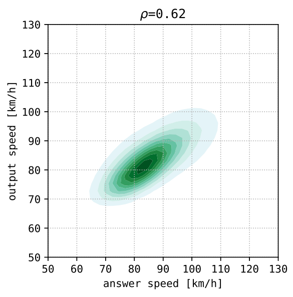


(a) LSG2C.

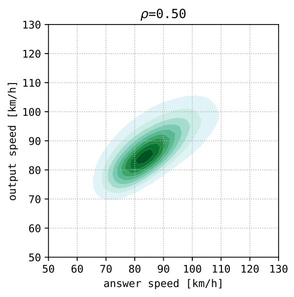


(b) LSG4C.

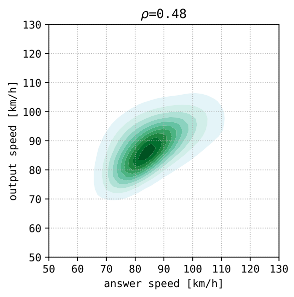
Figure 5.26: Validation of vehicle detection using sensors on lower flanges.



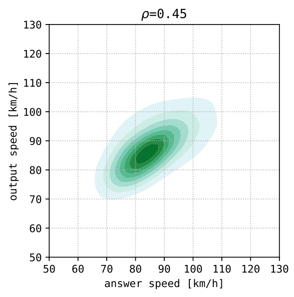
(a) RG2A1.



(b) RG2A2.



(c) RG4A1.



(d) RG4A2.

Figure 5.27: Validation of vehicle detection using sensors on girder supports.

In contrast, the correlation coefficients for the remaining three sensors were lower than 0.5. As seen in Fig. 5.23 (b), the performance of RG2A1 was not higher than other sensor types. We recommend using sensors on vertical stiffeners and lower flanges for the specific case of Bridge S.

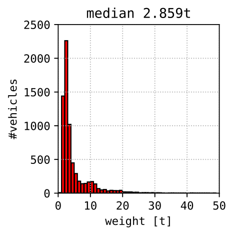
5.5 Single-Sensor BWIM

In this section, we demonstrate a BWIM system using only a single strain sensor based on the deep sensing proposal. On midnight from November 30 to 31, 2017, we performed test runs to obtain influence lines on Bridge C. We hired a dump truck (Isuzu GIGA) and made 16 round trips (32 traversals) on Bridge C. The truck had 10 wheels and three axles, and the axle weights were 3.44 t, 8.09 t, and 8.09 t from the front to the rear, respectively. After the test runs, we estimated 32 influence lines for strain sensor S1P4 by using the algorithm described in Section 2.4 proposed by Tateishi et al. [89].

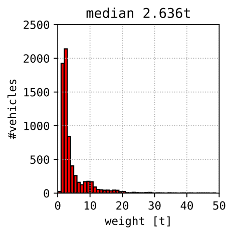
Fig. 5.28 shows the vehicle weights estimated using 15 influence lines. We ignored the fifth influence line because the observation data was lost as the result of a sensing system failure. The estimation was performed for LtoR vehicles that passed Bridge C from 08:00 to 16:00 on November 31, 2017. The mechanism was as follows. First, we obtained the score of vehicle existence for every second by Task 1 using Model 3. Next, we predicted the traveling speed and number of axles for every time slot whose sigmoid score was higher than 0.5. The axle positions could be estimated by peak detection from strain samples considering the number of axles obtained via deep sensing. Lastly, we estimated the sum of axle weights via the main process of BWIM following Eq. (2.4). In the strict sense, the label *vehicles* in the vertical axis of the graphs is the number of time slots when vehicles were detected. The locus information was not utilized in these experiments.

It should be noted that the deep sensing model was trained individually for each influence line, although the model could be shared with other influence lines. This sharing potential demonstrates the repeatability of our deep sensing proposal on Bridge C. The training and validation data included all vehicles in DS601, which were divided randomly into half for training and evaluation data. Through the experiments, we found that Model 3 frequently caused strong overfitting for the enlarged datasets. Therefore, we inserted an additional L2 normalization between the two linear layers shown in Fig. 5.2 to suppress overfitting. The network was accelerated by an NVIDIA GeForce GTX 1080 Ti GPU using CUDA [60] 9.2.

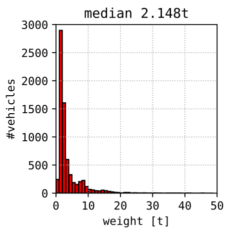
Through careful investigation of the estimated axle weights, we found a fault where vehicles were estimated to be much lighter than the ground truth. Such a case might be observed when the false peaks were detected and did not fit the true axle positions. To overcome this problem, we should include a fifth task of axle-position detection into the deep sensing models. Alternatively, we may create a fully neural BWIM system that does not use influence lines explicitly obtained by test runs. This system may be realized by installing an axle load meter and collecting ground-truth axle weights during the system initialization period.



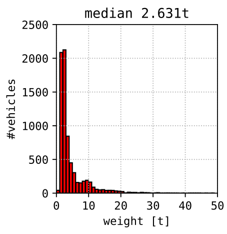
(a) Influence line 00.



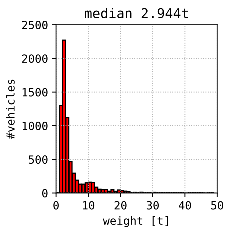
(b) Influence line 01.



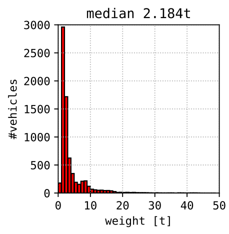
(c) Influence line 02.



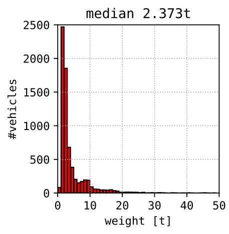
(d) Influence line 03.



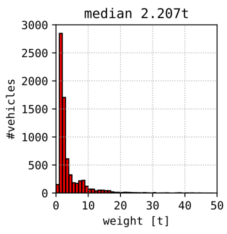
(e) Influence line 04.



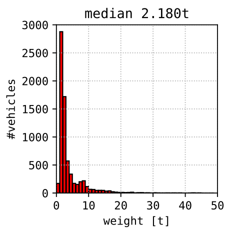
(f) Influence line 06.



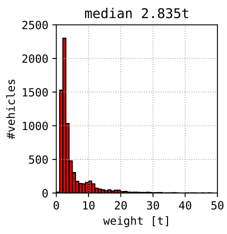
(g) Influence line 07.



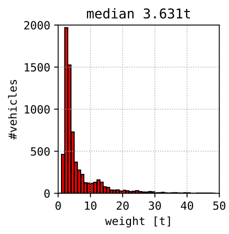
(h) Influence line 08.



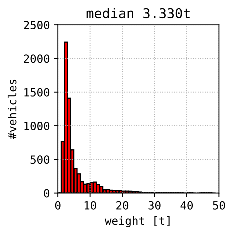
(i) Influence line 09.



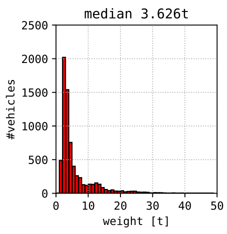
(j) Influence line 10.



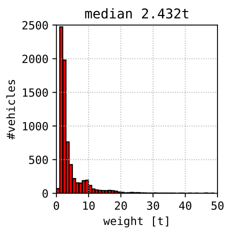
(k) Influence line 11.



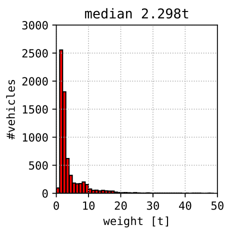
(l) Influence line 12.



(m) Influence line 13.



(n) Influence line 14.



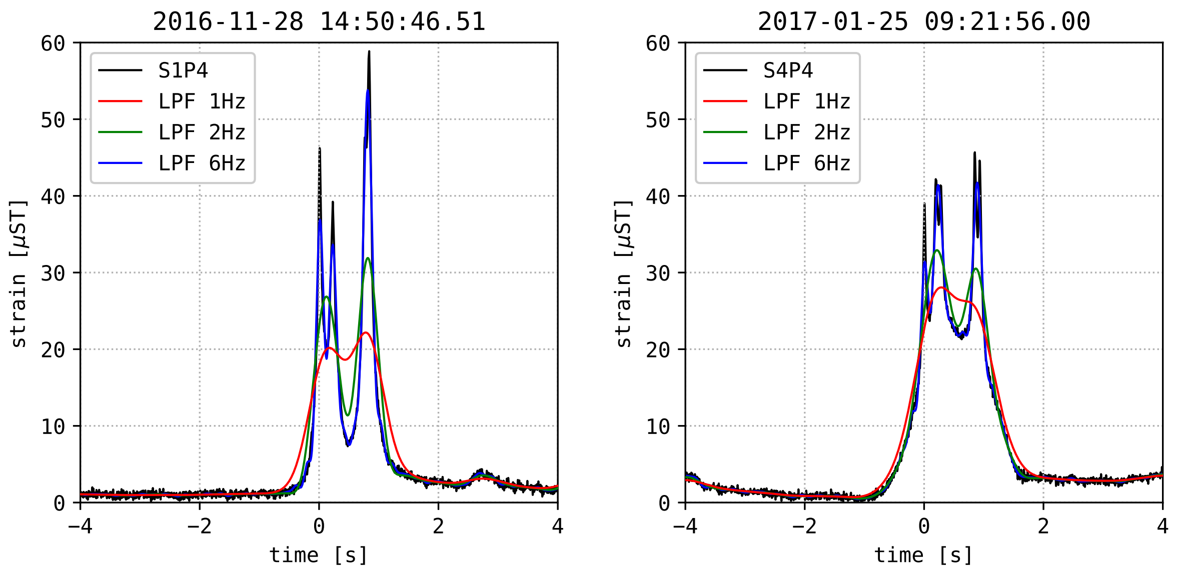
(o) Influence line 15.

Figure 5.28: Estimated LtoR vehicle weights on October 31, 2017.

5.6 Discussion

The major question in this work is the nature of the features in single-sensor signal data that are sufficient to specify vehicle properties with moderate accuracy. It is difficult to clarify the prediction mechanism in the deep CNNs, but we could identify which frequency components in the strain and vibration signals have significant effects on vehicle detectability. As shown in Table 5.5, even low-frequency components (under 10 Hz) contain sufficient information about individual vehicles for the strain and vibration signals. We can conclude that the proposed models did not focus mainly on the details of the signals but on the outline of the responses.

For the strain signals, we obtained Fig. 5.29 by applying an LPF to a strain data as a heavy vehicle crossed Bridge C. The large peaks in Fig. 5.29 indicate the number of axles or, more precisely, the number of shocks caused by the moving loads on the bridge. After applying an LPF with a cutoff frequency of 6 Hz, the resultant waveform fitted the original strain signal. By contrast, after applying a lower-frequency LPF, the peaks were less apparent, as shown in



(a) An LtoR vehicle.

(b) An RtoL vehicle.

Figure 5.29: Example of strain responses caused by vehicles after applying LPF.

Fig. 5.29. This may account for the decreased accuracy of Task 4 when a low-frequency LPF was used.

By contrast, the locus estimation Task 3 is inherently more difficult. In general, heavier vehicles bend the bridge deck more than lighter vehicles. The strain amplitude is also increased by the distance between the closest approach of the wheel axle to the strain meter, which is relative to the locus. The speed estimation Task 2 is also inherently difficult because the video timestamps were shifted randomly, as described in Section 5.2. Consequently, naïve statistical values such as amplitude and peak positions explain little about the regression tasks.

Although the correct answer to the major question has yet to be revealed, we have developed a hypothesis. We infer that the deep sensing models learned the influence surface for the 2-dimensional road surface of Bridge C. As we described in Eq. (2.4), a typical influence line is defined as a 1-dimensional curve $i(x)$. However, we can also assume an influence surface $i(x, y)$ where y indicates a position in the direction orthogonal to the bridge axis. The vehicle speed $\dot{x}$ and locus $\overline{y}$ can then be predicted from $i(x, y)$.

As described in Section 5.4, the sensor installation point may have a great influence on the estimation accuracy. To optimize this accuracy, we should first seek the best installation point and then prepare the ground-truth data as accurately as possible. For Bridge S, we adopted a primitive approach based on peak detection for acquiring ground-truth data. Some vehicles were ignored because their estimated speed appeared to be too fast or too slow. Such a situation can occur frequently, particularly when multiple vehicles, running in different lanes, pass over the sensors almost at the same time. The primitive peak-based approach that we employed is not always able to discriminate between individual vehicles in such a situation. This may be one of the reasons why the accuracy achieved was inadequate for practical applications. To improve the dataset accuracy, we should utilize a radar speed gun or a high-resolution camera instead of relying on the naïve peak-based method.

For the accelerometers A3XP8, A3YP8 and A3ZP8, the performance of the axle counting Task 4 was much lower than those for strain sensors. This decreased performance may be because identifying individual moving axles from the acceleration graphs shown in Fig. 4.3 was much more difficult than identifying those from the strain graphs in Fig. 4.2. As for Bridge C, the acceleration signals had poor sensitivity or spatiotemporal locality of the individual moving axles. At the time, A3ZP8 was installed at the bottom of the box girder. To improve the axle detectability from acceleration signals, we should install the sensor closely underneath the bridge deck to capture vibrations caused by individual axles.

5.7 Conclusion

In Chapter 5, we have proposed a *deep sensing* approach to a single-sensor BWIM system that detects target vehicles using only a single strain or vibration sensor. The vehicle properties obtained by the proposed CNNs include lane, speed, locus, and axles, which are useful for BWIM systems. The novelty of our work is in its demonstration of the information richness of the single-sensor data by showing that speed and locus estimation is possible without using multiple sensors installed at multiple positions on the bridge. The single-sensor-based detector may simplify conventional BWIM systems dramatically so they can be applied more widely to bridges at only moderate cost.

The training data for the proposed vehicle detector can be obtained by analyzing traffic surveillance videos. The training data may contain nonideal, congested traffic conditions, with a vehicle appearing every second. The model parameters are optimized statistically without requiring any heuristics, except for the length of the sensor-data sequences fed to the models. This optimization is one of the advantages over heuristic vehicle detectors, enabling the systems to be installed on many other bridges in the future. We also believe our proposal can be merged with the weighing process in a CNN after we have collected substantial amounts of vehicle weight data.

Our current task is to examine the effectiveness of the deep sensing proposal with other bridges and for other sensors such as displacement sensors. In addition, we plan to integrate the weighing process into the deep sensing models by collecting vehicle axle load data and exploiting the obtained properties, particularly the loci, in the weighing process. Moreover, we should compare the performance, under the same conditions, of our proposal with that for conventional vehicle detectors that use peak detection. To complete these tasks, we may need to add extra sensors to the bridges.

Chapter 6

Anomaly Strain Detection

In this chapter, we present an anomaly detection approach for bridge damage detection that consults a traffic surveillance camera on the bridge and a single strain sensor underneath the bridge deck. As described in Section 2.4, the strain response on bridge components can be modeled by a linear response model. Although the linearity may not hold for all vehicles and bridges, we assume the existence of a response model that may explain the strain response against passing vehicles. Once the response model was obtained, the strain response may be predictable by feeding the vehicle properties to the response model. If we set up such a predictor during the bridge's construction, we could capture small signs of deterioration later by comparing predicted waveforms and real observation data. Fortunately, vehicle properties including shape and motion but excluding axle loads is obtainable using a surveillance camera. However, we encountered a difficulty, namely that the video data could not supply information about a vehicle's axle weights. As described in Section 2.4, the strain response depends on the moving axle loads, so measuring axle weights directly seems an obvious approach. To obtain the axle weights, the options are to install a pavement sensor or use BWIM systems [47, 99]. As discussed in Section 2.4, the former is fragile, difficult to retrofit to existing bridges, and limits the traveling speeds. In contrast, the latter requires that strain response characteristics be obtained in advance, which is not applicable to the anomaly detection problem. We therefore abandoned the axle weight approach.

Instead, we developed a new sensor-fusion approach that directly compares the vehicle image and strain response in a common feature space. Fig. 6.1 shows an architecture of the proposed system, named the early warning system (EWS). The fact that the deep sensing models in Chapter 5 successfully extracted vehicle speeds and loci even though the strain waveforms were shifted deliberately at random indicates the possibility of learning an inverse response function. If the hypothesis is true, the bridge damage may be detectable by comparing vehicle properties estimated by deep sensing and video analysis. Once a bridge becomes damaged and the response model changes, the features of a target vehicle may differ between video and sensor data. The feature comparison is performed for every passing vehicle. Some video–response pairs caused by the same vehicle may be inconsistent in the common space. We treat such an event as an *anomaly*, making the assumption that such a response may be caused by bridge structural damage.

This approach is somewhat similar to the Siamese network [5], which compares a pair of images via features extracted by the same neural network and calculates a similarity score. However, our approach utilizes two different neural networks for the video and strain data. Moreover, unlike the Siamese network, the two networks are trained to predict vehicle speeds and loci individually. As demonstrated in Chapter 5, vehicle speeds and loci may be predictable from both video and strain data, and they affect the shape of the strain response significantly.

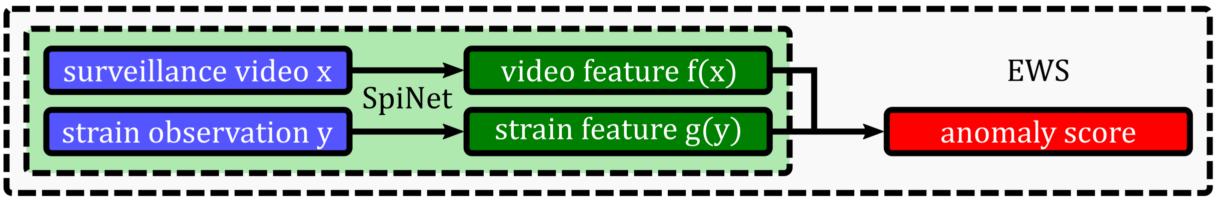


Figure 6.1: System architecture for the proposed anomaly detection system.

By learning these two tasks, the two networks can acquire feature spaces that seem to comprise *common factors* of the video and strain data. Finally, the bases of the two spaces can be matched by minimizing the distance between two related elements in the respective spaces. We call this approach *spiral learning*. It should be noted that bad weather, e.g., a snowstorm or heavy rain, may disrupt the video signal. In such a situation, an event may be misidentified as an anomaly even if the strain response was normal. Therefore, we proposed adding an adversarial learning mechanism as a countermeasure.

6.1 Anomaly Detection Network

Spiral learning is a feature-matching technique whereby a pair of samples, which are observed by two sensors but share the same latent variable, are fed into two separate networks. The networks are independent of each other, except for a final linear layer that shares the same weight matrix. Each network learns from its respective dataset and acquires a feature mapping that contains the shared latent variable. In this dissertation, the latent variable indicates the properties of every passing vehicle, such as traveling speed and locus.

Fig. 6.2 shows the proposed neural network architectures. The network, named the *spiral network*, consists of two CNNs, namely the *CamNet* and the *SigNet*.

The CamNet receives video data observed by a traffic surveillance camera and outputs the video features for every target vehicle. The input is 50 grayscale video frames (taken over two seconds) recorded from when the vehicle enters the camera's field of view. Each frame is resized to 224×224 pixels in advance. The network was designed in reference to VGG16 [84], except for using preactivated residual blocks [22] instead of plain convolution layers.

The SigNet receives signal data observed by a strain sensor underneath the bridge deck and outputs the signal features for every target vehicle. The input is four-second batches of raw strain data sampled at 200 Hz. Each strain sample was rescaled so its maximum and minimum were normalized to 1 and 0, respectively, for effective learning. The network was designed by increasing the number of residual blocks of the deep sensing models proposed in Chapter 5.

The outputs of the two networks are fed to a fully connected layer mutually. This linear layer estimates speed and locus of the target vehicle using either video or signal features.

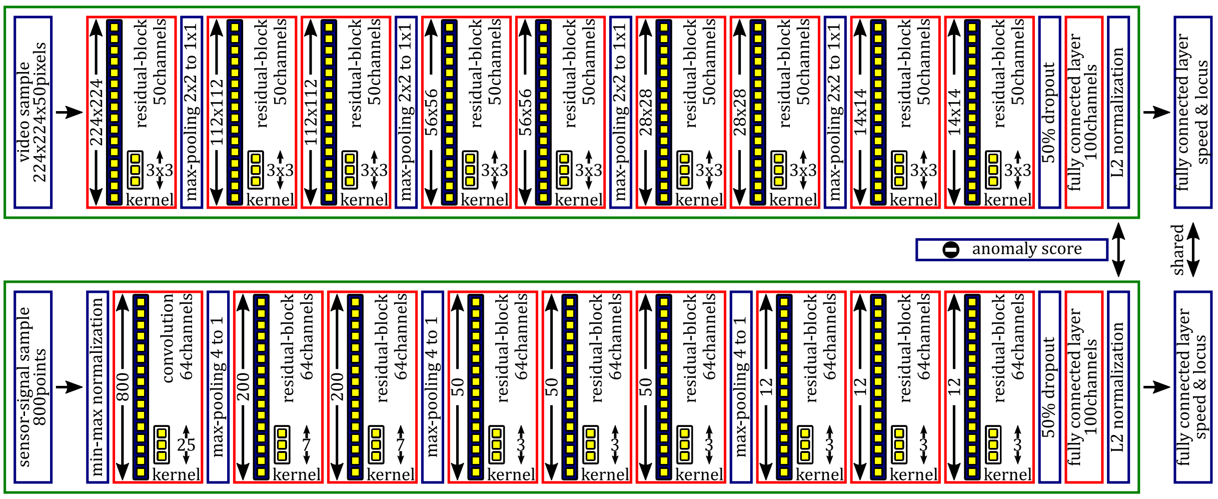

video sample 224x224x50pixels
224x224
residual-block 50channels
3x3 kernel
max-pooling 2x2 to 1x1
112x112
residual-block 50channels
3x3 kernel
112x112
residual-block 50channels
3x3 kernel
max-pooling 2x2 to 1x1
56x56
residual-block 50channels
3x3 kernel
56x56
residual-block 50channels
3x3 kernel
max-pooling 2x2 to 1x1
28x28
residual-block 50channels
3x3 kernel
28x28
residual-block 50channels
3x3 kernel
max-pooling 2x2 to 1x1
14x14
residual-block 50channels
3x3 kernel
14x14
residual-block 50channels
3x3 kernel
50% dropout
fully connected layer 100channels
L2 normalization
fully connected layer speed & locus
anomaly score
shared
sensor-signal sample 800points
min-max normalization
800
convolution 64channels
25 kernel
max-pooling 4 to 1
200
residual-block 64channels
7 kernel
200
residual-block 64channels
7 kernel
max-pooling 4 to 1
50
residual-block 64channels
3 kernel
50
residual-block 64channels
3 kernel
50
residual-block 64channels
3 kernel
max-pooling 4 to 1
12
residual-block 64channels
3 kernel
12
residual-block 64channels
3 kernel
12
residual-block 64channels
3 kernel
50% dropout
fully connected layer 100channels
L2 normalization
fully connected layer speed & locus

Figure 6.2: Network architectures for video and strain data.

Note that we applied a *third* activation function to the output of the residual blocks in addition to the two activation functions inside the blocks. We used ReLU [19] for the activation functions in each layer of the two networks, except for the output layers. To suppress overfitting, we inserted a 50% dropout [86] before each linear layer in each network.

6.1.1 Plain Spiral Learning

By learning the two tasks of predicting vehicle properties, the two networks for video and sensor data can acquire feature spaces that seem to comprise common factors of the video and sensor data. We selected speed and locus as the outputs. As described in Section 2.4, these two properties have strong effects on the signal shape of the strain response. We can expect the two networks to generate the same feature vector as a common factor through learning the two prediction tasks and sharing the same output layer.

We exploit a multitask learning (MTL) technique [9] for the deep sensing models described in Chapter 5. The loss function $\mathcal{L}$ for the CamNet is defined in Eq. (6.1).

$$\mathcal{L}(f,h) = \frac{1}{N}\left\{\sum_{n=1}^{N}\frac{\left[h_{\mathrm{s}}(f(\boldsymbol{x}_n)) - s_n\right]^2}{\mathrm{Var}(s)} + \sum_{n=1}^{N}\frac{\left[h_{\mathrm{l}}(f(\boldsymbol{x}_n)) - l_n\right]^2}{\mathrm{Var}(l)},\right\} \tag{6.1}$$

where $\boldsymbol{x}_n$ is the n-th video sample. f and h denote the feature extraction through the CamNet and the final linear layer shared by the two networks, respectively. s and l are ground-truth annotations for the speed- and locus-prediction tasks, respectively. It should be noted that $f(\boldsymbol{x})$ was normalized as shown in Fig. 6.2.

The loss function $\mathcal{L}$ for the SigNet is defined in Eq. (6.2).

$$\mathcal{L}(g,h) = \frac{1}{N}\left\{\sum_{n=1}^{N}\frac{\left[h_{\mathrm{s}}(g(\boldsymbol{y}_n)) - s_n\right]^2}{\mathrm{Var}(s)} + \sum_{n=1}^{N}\frac{\left[h_{\mathrm{l}}(g(\boldsymbol{y}_n)) - l_n\right]^2}{\mathrm{Var}(l)}\right\}, \tag{6.2}$$

where $\boldsymbol{y}_n$ is the n-th signal sequence and g denotes the feature extraction through the SigNet. It should be noted that $g(\boldsymbol{x})$ was normalized, as shown in Fig. 6.2.

The final linear layer is shared by the two models, enabling the outputs of the two networks to be treated as feature vectors in a common feature space. Both networks learn the correlation between the two sources by drawing their feature vectors together. The attracting mechanism can be described as the anomaly loss $\mathcal{A}$ defined in Eq. (6.3).

$$\mathcal{A}(f, g) = \frac{1}{N} \sum_{n=1}^{N} \| f(\boldsymbol{x}_n) - g(\boldsymbol{y}_n) \|_2^2 . \tag{6.3}$$

Consequently, the optimization problem for the combined network, named the *SpiNet*, can be described in terms of multitasking [9], as given in Eq. (6.4).

$$\mathcal{L}_{\mathrm{MSE}}(f, g, h) = \mathcal{L}(f, h) + \mathcal{L}(g, h) + \lambda \mathcal{A}(f, g), \tag{6.4}$$

where λ is a weight of $\mathcal{A}$ and is set as 10. Eq. (6.4) minimizes five individual losses, namely speed and locus prediction from the video data, speed and locus prediction from the strain data, and the L^2 norm between the feature vectors for video and sensor data. Because these two networks share the output layer, Eq. (6.3) plays the role of matching the correlative elements from the two feature spaces. As a result, the video and strain feature spaces will coalesce after a long training period as two *cannibal black holes* forming a spiral trajectory.

The video and strain data may differ from each other, although they cover the same target vehicle. This difference can be identified by monitoring the L^2 norm of the subtraction between $f(\boldsymbol{x})$ and $g(\boldsymbol{y})$. We therefore define the L^2 norm as an *anomaly score*. The outliers of the distribution are identified as anomalous vehicles.

6.1.2 Adversarial Spiral Learning

Anomalous vehicle detection based on spiral learning depends on features extracted from the surveillance video recordings. The video can be disturbed by environmental conditions such as weather, traffic jams, pedestrians, and vehicles in the opposite lane. Such disturbances may cause mistaken anomaly detections. The *adversarial mechanism* can reduce these errors, which may enable video features to correlate less with the weather conditions, including heavy rain, snowstorms, deep snow, and morning haze. We therefore combined the adversarial learning concept introduced in Section 3.2 with our spiral learning proposal for this purpose. Fig. 6.3 illustrates the mechanism.

One of the simplest methods to achieve weather resistance is to use a weather discriminator. The discriminator tries to find videos recorded under bad weather conditions by examining video features carefully and in detail. To implement this function, we need to append weather

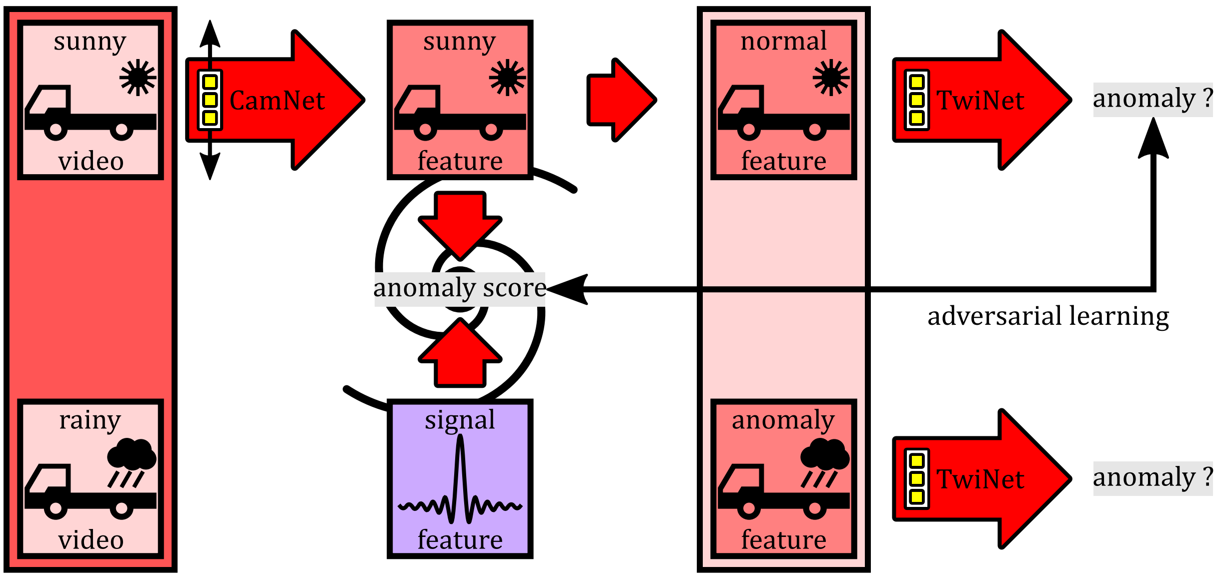


Figure 6.3: Adversarial spiral learning mechanism.

tags to the traffic dataset. The loss function for the SpiNet can be described as in Eq. (6.5), using the mean cross entropy $\mathcal{L}_{\text{CE}}$:

$$\mathcal{L}_{\text{GAN}}(f, g, h) = \mathcal{L}_{\text{MSE}}(f, g, h) - \mathcal{L}_{\text{CE}}(p, q), \tag{6.5}$$

where p and q denote the discriminator and the weather tag, respectively. After a long training period, the discriminator can no longer find faults in the obtained video features.

Initially, we tried to tag each vehicle by consulting the historical climate data archived by the government, but we encountered a major difficulty. The weather conditions at the bridge did not always correspond to the historical data, because weather conditions were recorded at the nearest observation station, a few kilometers away from the target bridge. We then tried to tag the vehicles manually by watching the video, but we encountered another difficulty. Because of the complicated weather situations, it was difficult to formulate a robust policy for weather annotation. For example, should there be a cloudy tag in addition to a sunny tag, and is there a boundary between cloudy and light-rain conditions, or between snowfall and snow accumulation? Sometimes, the video lost focus because of morning haze, twilight, or a fogged lens. Additionally, precipitation and the intensity of solar radiation, which also has a strong impact on the video quality, should be noted.

We therefore abandoned the weather annotation plan and developed a fully unsupervised approach. Here, the issue was that a vehicle might be mistakenly judged to be anomalous because of bad weather, even though the strain response was normal. This issue was a situation where an *anomalous strain response* could be found without consulting the strain feature, but by consulting the video features alone instead. This might be a problematic situation based on the main purpose of the strain characteristics analysis. We therefore defined the ground-truth tag for the discriminator, as given in Eq. (6.6).

$$q(\boldsymbol{x}, \boldsymbol{y}, f, g) = \mathbb{H}\left\{\|f(\boldsymbol{x}) - g(\boldsymbol{y})\|_2^2 - \mathcal{A}(f, g)\right\}, \tag{6.6}$$

where $\mathbb{H}$ is the step function. The initial value of $\mathcal{A}$ for the first epoch was 0.1. The adversarial network, named the *TwiNet*, was defined as a perceptron whose hidden layer has 10 dimensions.

In later sections, the adversarial version of the spiral learning is called the GAN model, and the primitive version described in Section 6.1.1 is called the mean squared error (MSE) model.

6.2 Training and Evaluation Data

In Section 6.3, we demonstrate our proposal for the case of Bridge C. As shown in Fig. 4.1 (a), we installed a traffic surveillance camera and some strain sensors on the bridge. The camera was installed at the bridge entrance to capture images of vehicles in close proximity to the vehicle bodies. We also deployed four highly sensitive strain sensors underneath the bridge deck to collect strain responses in the direction orthogonal to the bridge axis. The sensors are shown as the four red triangles denoted S1P4, S2P4, S3P4, and S4P4, respectively, in Fig. 4.1. All of the sensors were synchronized except for the camera, and their sampling rate was 200 Hz. The frame rate of the camera was set to 25 frames per second.

In preparation for the experiments, we needed ground-truth data of target vehicles on Bridge C. We utilized the dataset DS601 prepared in Chapter 4 for this purpose. DS601 was created via a traffic surveillance system (TSS) [32] based on the Faster R-CNN [73] and contains information about 996,093 vehicles that crossed the bridge between 08:00 and 16:00 from November 5, 2016, to April 28, 2017. In this work, we ignored all vehicles with two axles because civil engineers are mainly interested in heavy vehicles.

We then collected videos of the vehicles and sensor signals caused by the vehicles. The n-th vehicle record is described as a triplet $\{\boldsymbol{x}_n, \boldsymbol{y}_n, \boldsymbol{t}_n\}$. The video input $\boldsymbol{x}_n$ was a sequence of length 50. The length of signal input $\boldsymbol{y}_n$ was set as 800 (4 s), which is half the input length for the deep sensing models in Chapter 5. This setting aimed to remove the effect of following vehicles. $\boldsymbol{t}$ is the ground truth of the vehicle speed and locus.

We prepared the *trainval* and *evaluation* datasets by randomly dividing DS601 in half. Then, 80% of the *trainval* vehicles were assigned as *training* data, with the remaining 20% being *validation* data. The validation data were utilized for early stopping [98], where each time $\mathcal{L}_{\mathrm{MSE}}$ for the validation dataset reached a new minimum, the model was saved. The evaluation processes were performed with models that updated the minimum of the validation loss in the same fashion as the early stopping approach [98].

For LtoR vehicles, the *trainval* and *evaluation* datasets contained 17,757 and 17,967 vehicles, respectively. For RtoL vehicles, the datasets contained 20,996 and 21,078 vehicles, respectively.

Table 6.1: List of stormy days (DS102).

2016-12-06	2016-12-10	2016-12-11	2016-12-14	2016-12-16	2016-12-24
2016-12-27	2016-12-28	2017-01-05	2017-01-17	2017-01-19	2017-01-21
2017-01-23	2017-01-24	2017-01-25	2017-02-07	2017-02-21	2017-03-07
2017-03-24	2017-03-27	2017-04-09	2017-04-12		

6.3 Experimental Results

The proposed networks were written in Python 2.7 and implemented on Chainer [68] 5.0.0. They were accelerated by an NVIDIA GeForce GTX 1080 Ti and CUDA [60] 9.2. We employed the AMSGrad [72] as an implementation of stochastic gradient descent, whose parameters were set as defined in the Chainer implementation by default. The mini-batch size was set at 10.

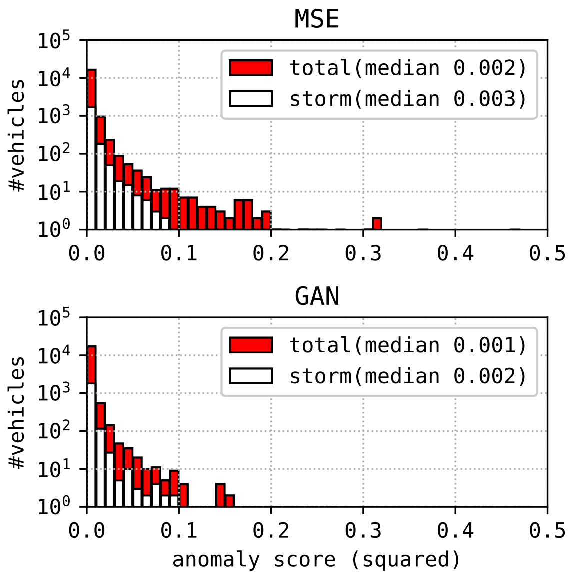
The combined network was evaluated mainly in terms of the final generalization performance after loss convergence and before overfitting. The performance was visualized using histograms for anomaly score and waveforms of high anomaly scores. As described in Section 6.2, the evaluation processes were performed with models whose validation losses were minimal in the same fashion as the early stopping approach [98]. It should be noted that the mechanism of early stopping was corrected, which had not worked correctly in the previous version [33].

6.3.1 Anomaly Score Distribution

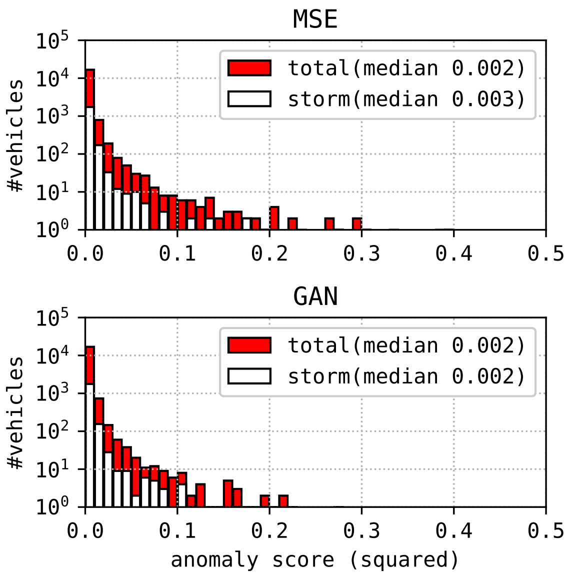
We evaluated the distribution of anomaly scores by drawing histograms. We compared two cases, namely the MSE and GAN models. We also drew histograms for stormy days with heavy snow or haze for several hours to validate the robustness of the anomaly score against weather conditions. Table 6.1 lists such stormy days found by examining whether the video clearly seemed to involve snowy or hazy conditions, paying attention to the video sharpness.

Fig. 6.4 shows the logarithmic histograms of the anomaly scores estimated for LtoR vehicles. Fig. 6.5 shows the histograms for RtoL vehicles. The anomaly scores were estimated using strain sensors S1P4 and S2P4 for LtoR vehicles and S3P4 and S4P4 for RtoL vehicles. As shown in Fig. 4.1 (a), these sensors were installed to be under the wheels of LtoR or RtoL vehicles. As we anticipated, most anomaly scores were distributed around zero, and a small number of responses were identified as anomalous. In Chapter 6, we do not consider the proper thresholds for anomaly detection.

Focusing on the stormy days, the median score was a little higher than the total distribution for both plain and adversarial cases, although the population was much smaller than the

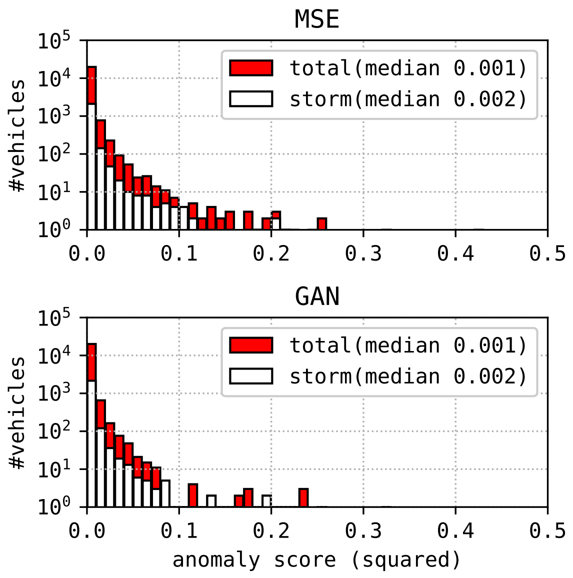


(a) S1P4.

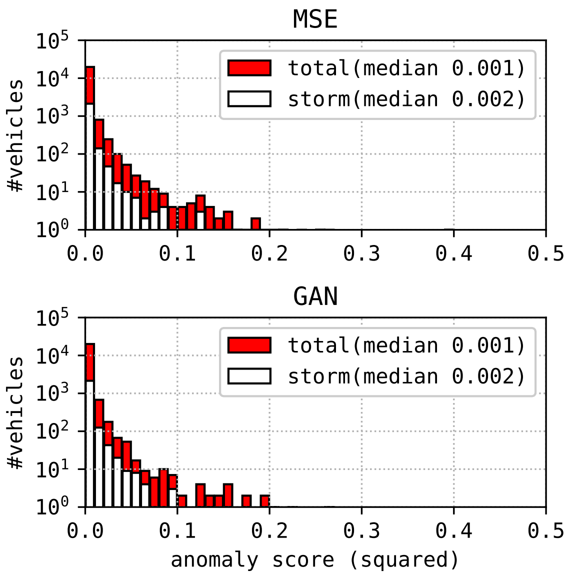


(b) S2P4.

Figure 6.4: Anomaly score distributions estimated for LtoR vehicles.



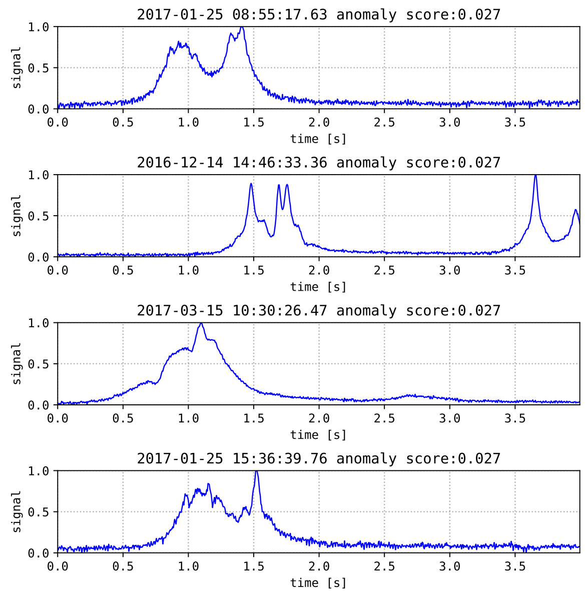
(a) S3P4.



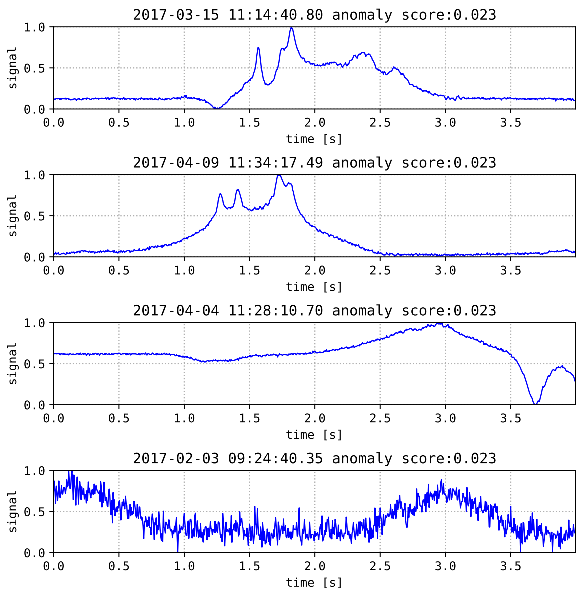
(b) S4P4.

Figure 6.5: Anomaly score distributions estimated for RtoL vehicles.

total. Unfortunately, we found no remarkable difference between the distributions of plain and adversarial cases. Consequently, the video features have trivial correlation with the anomaly score, and thus the robustness against changing weather conditions was able to be obtained via the MSE model. It should be noted that the DS601 database involves both sunny and snowy days, and the SpiNet might be able to obtain weatherproof results without the adversarial

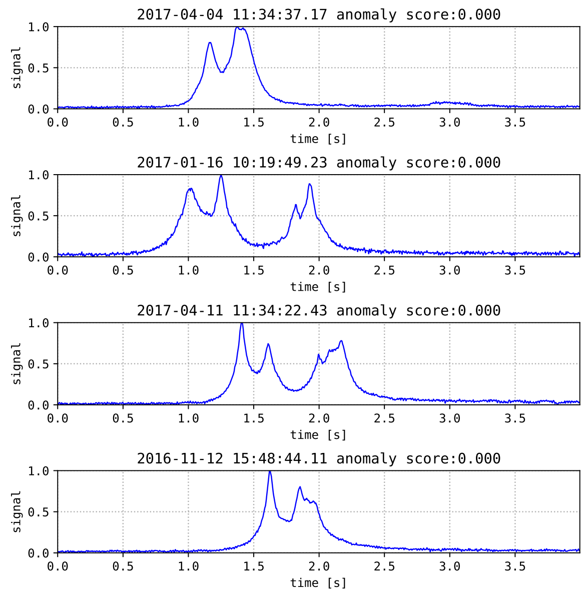


(a) LtoR (S1P4).

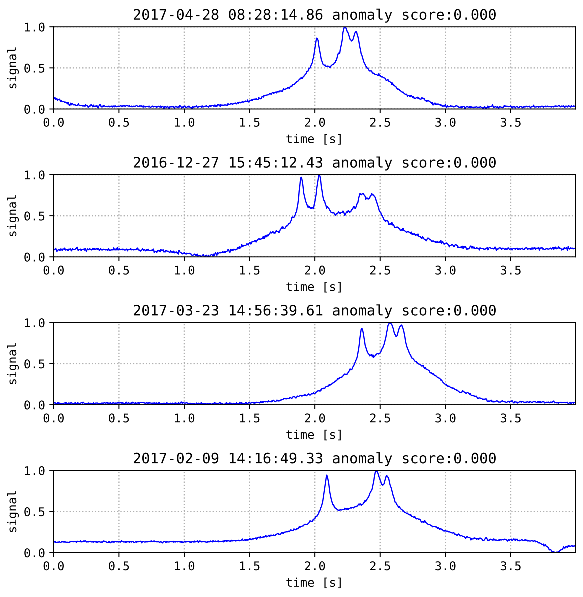


(b) RtoL (S4P4).

Figure 6.6: Top 2% anomalies detected by the MSE model.



(a) LtoR (S1P4).



(b) RtoL (S4P4).

Figure 6.7: Bottom 2% anomalies detected by the MSE model.

method.

6.3.2 Anomaly Response Examples

We examined each anomalous and normal strain response detected by the MSE model. Fig. 6.6 shows four anomalous responses for the top 2% for LtoR and RtoL vehicles passing over strain sensors S1P4 and S4P4, respectively. Fig. 6.7 shows four normal responses for the bottom 2% in terms of anomaly score. In the graphs, the sensor signals were normalized so that their maximum and minimum were 1 and 0, respectively.

Then, we investigated the top 50 anomalous responses in detail [33] and classified them into four classes as follows. It should be noted that the true meaning of the anomaly score was not revealed, and the following interpretation was only a hypothesis. However, we consider that the SpiNet may indicate high anomaly scores because of some abnormalities with regard to vehicle bodies and traffic conditions, even though the bridge structure itself is healthy.

Traffic Jam Some extremely slow vehicles may be misidentified as anomalies, in particular under snowstorm conditions. In such a situation, a vehicle takes an excessive amount of time to arrive at the sensor installation position, exceeding the four seconds allowed. The SigNet will therefore fail to capture the strain peak caused by the vehicle. We also found cases where the SigNet identified false peaks caused by other vehicles appearing just before the target vehicle arrived at the bridge entrance. To deal with such extremely slow vehicles, the input length of the SigNet should be extended to enable the capture of the peaks for all targets. It should be also noted that a vehicle caught in a traffic jam might not be able to drive at a steady speed, which may also affect the strain response.

Multiple Vehicles Some vehicles may be misidentified as anomalies when another heavy vehicle passes the sensor at almost the same time, causing a mixture of strain responses caused by multiple individual vehicles. Especially if two vehicles running in opposite lanes pass the sensor at the same time, the mixed strain signal may contain two groups of influence lines whose shapes are completely opposite.

Cargo Vibration In some vehicles, cargo may be moving about, with the resulting mechanical shock being captured by the strain sensor. This movement may cause ripples in the strain peak, causing the signal to resemble that of a vehicle with additional wheels. Such a situation may be observed, in particular with lightly loaded vehicles, but it was hardly detectable from an image with a resolution of 224×224. Because the road surface was flat, a heavily loaded cargo bed was unlikely to vibrate.

Dataset Error Some vehicles may be misidentified as anomalous because of errors in the traffic dataset. We found that some cars and motorbikes with two axles were included in the large-vehicle dataset and were then classified as anomalies. Some cars have more than two axles if they are towing trailers or other vehicles. In fact, the vehicles with the highest anomaly scores had poor signal–noise ratios and were almost buried in noise. By intuition, we believe that such vehicles may be identified as anomalous in this class not only because their appearance was rare, but also because of the noise.

6.3.3 Effect of Training Data Volume

For the experiments in Section 6.3, we collected data on vehicles that crossed Bridge C during a six-month period. It is preferable to use the smallest feasible dataset, from the viewpoint of reducing the training cost at system initialization. By contrast, as described in Section 6.3.1, we found no remarkable effect of the adversarial mechanism on the distribution of the anomaly score. Although there is a possibility that video features may have some correlation with bad weather conditions, the amount of training data volume may cover the correlation.

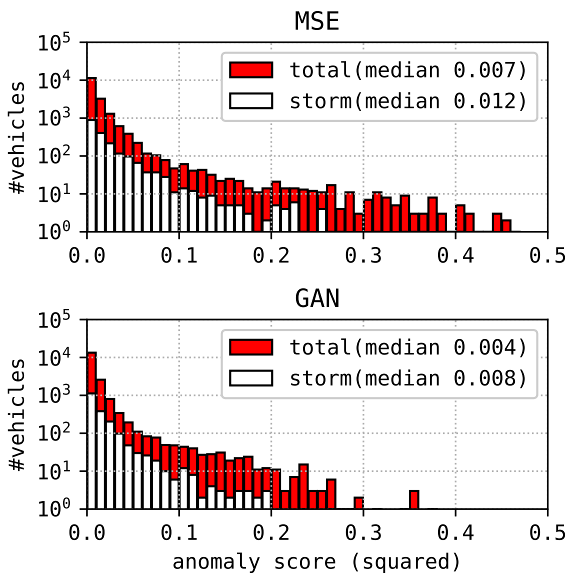
We thus conducted an additional experiment to investigate the contribution of dataset volume to score distribution. We subsampled each *trainval* dataset for training and validation so that the volume of sampled *trainval* data varied from 10% to 100%.

Fig. 6.8 shows the anomaly score distributions for LtoR vehicles in the evaluation dataset under the subsampling conditions. As shown in the graphs, the median anomaly score for all vehicles decreased as the dataset volume increased from 20% to 80%. At the same time, the median anomaly score for stormy days also decreased. Thus, the training data volume has a strong effect on the anomaly score distribution. Fig. 6.9 shows the distributions for RtoL vehicles. As for RtoL vehicles, the median anomaly score decreased as the dataset volume increased from 20% to 80%.

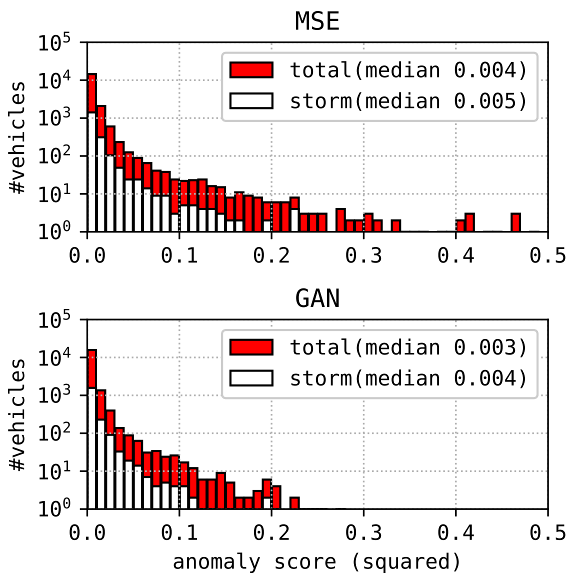
Focusing on the median anomaly score, we may confirm the small effect of the adversarial mechanism. Fig. 6.10 compares the effect of training data volume on the anomaly score distribution for the evaluation dataset for the cases of MSE and GAN model. Fig. 6.10 (a) shows the results for LtoR vehicles, while Fig. 6.10 (b) shows the results for RtoL vehicles. In our experiments, we found that the median anomaly score obtained by the GAN model was lower than that obtained by the MSE model. The effect of the adversarial mechanism on the median anomaly score became stronger with less training data.

6.4 Discussion

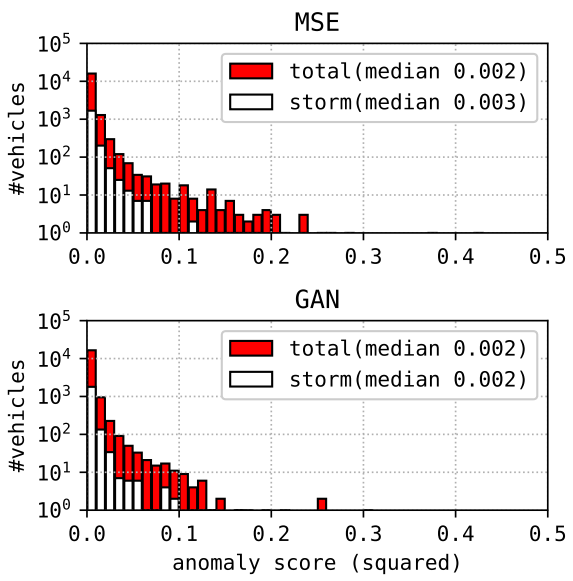
The main problem of this work was the interpretation of the anomaly score, which was not connected directly to the physical abnormality of the bridge structure. The bases of anomaly detection were feature vectors for video and strain signals, which were extracted via two CNNs.



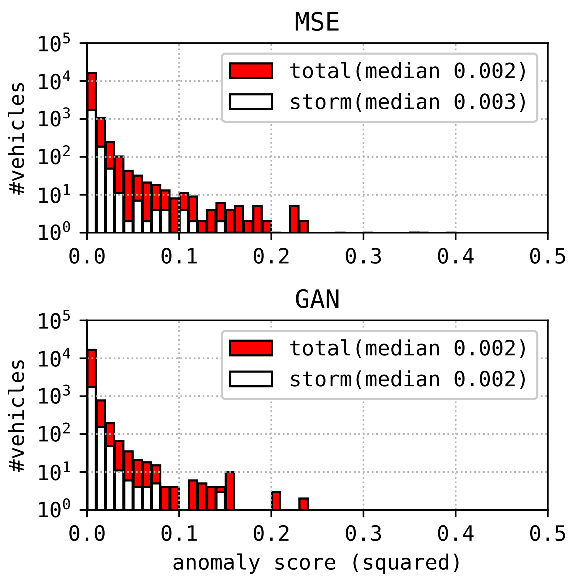
(a) 20%.



(b) 40%.



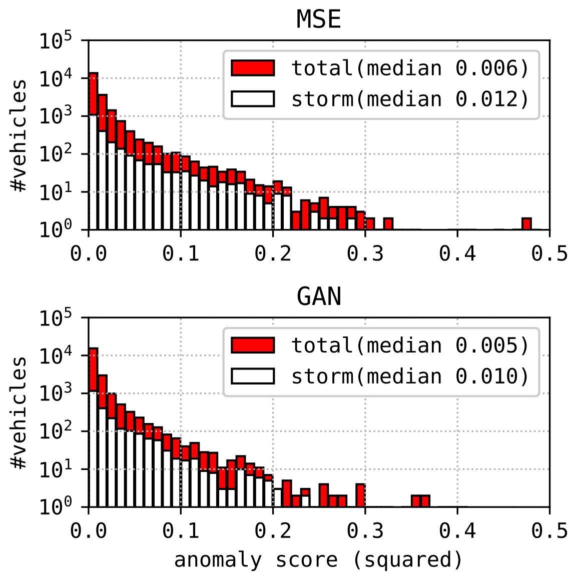
(c) 60%.



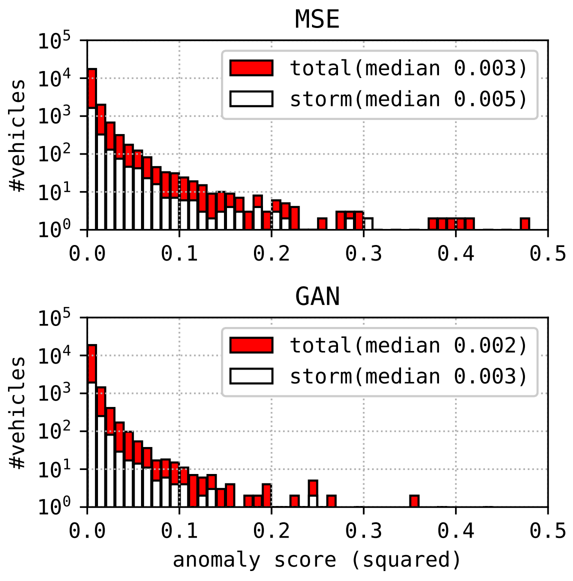
(d) 80%.

Figure 6.8: Effect of training data volume for LtoR vehicles (S1P4).

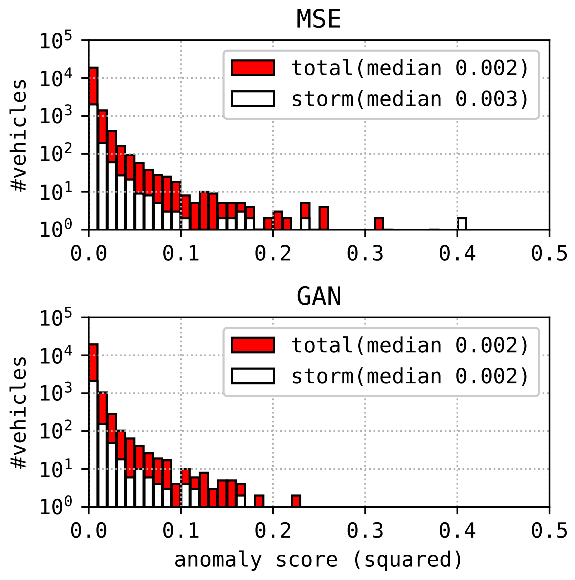
The meaning of the feature vectors was not completely explored, and the anomaly score may become high because of reasons that are irrelevant to the structural damage to the bridge, as explored in Section 6.3.2. Unfortunately, we had no collection of ground-truth data for bridge damage detection, and thus we could not explore which types of abnormality may be detectable or undetectable by the proposed system in a quantitative manner. In this sense, our experiments in Chapter 6 are ambitious and unfortunately, their effectiveness is currently uncertain. From the perspective of real application, a major weakness with regard to the anomaly scores is that



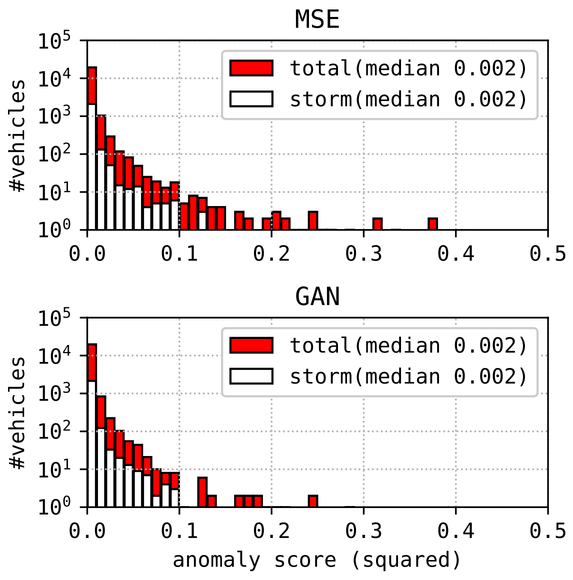
(a) 20%.



(b) 40%.



(c) 60%.

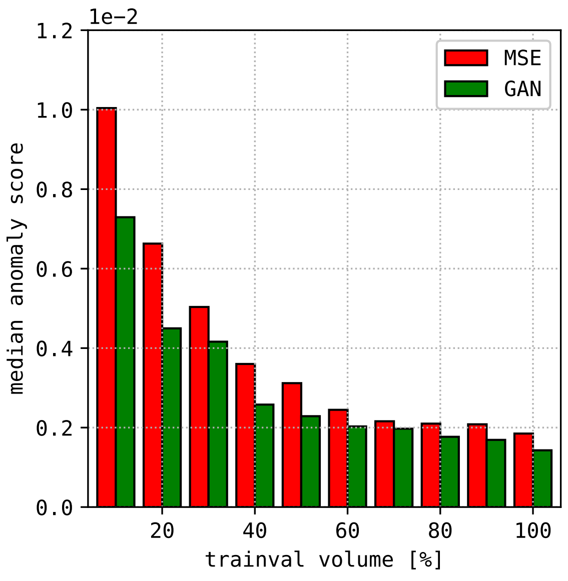


(d) 80%.

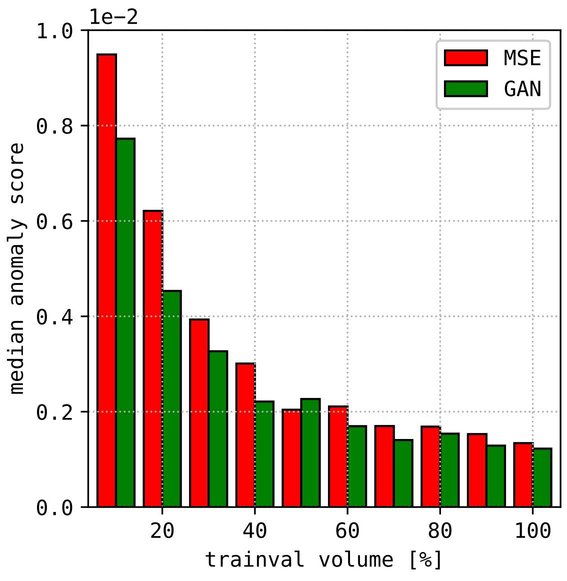
Figure 6.9: Effect of training data volume for RtoL vehicles (S4P4).

of identifying the causes of a large number of anomalies.

The anomalous response detection based on spiral learning depends on features extracted from the surveillance video and strain data obtained by a single strain sensor. Video features can be disturbed by environmental conditions, including weather, traffic jams, pedestrians, and other vehicles in the opposite lane. For Bridge C, pedestrians were rare, and we did not observe cases where a vehicle was mistakenly detected as anomalous because of pedestrians. Light vehicles in the other lane were also not a problem because the large traffic volume provided the



(a) LtoR.



(b) RtoL.

Figure 6.10: Effect of training data volume on the median anomaly score.

SpiNet with sufficient opportunities for learning such situations. Traffic jams may be caused by construction work, snowstorms, or traffic signals. For example, the far-side lane was closed on November 9, 2016, because of construction work. Under such conditions, the bridge behaved abnormally because many vehicles were required to use the open lane in the opposite direction to that normally used. We need not be concerned with such one-off events. However, traffic signals can be a major problem in general. This problem is not addressed in this dissertation, however, because there were no signals near our target bridge.

Another doubt about our results was the fact that some anomalies could be detected without consulting the video data. We are nevertheless confident that the video features are necessary as explanatory variables for strain analysis. However, there might be some anomalies for which the strain response appears strange at first glance, e.g., a strain response completely buried in white noise. Failures in sensors may also cause such situations. We need to investigate in detail what type of anomaly requires the spiral learning approach.

6.5 Conclusion

In this chapter, we have proposed a novel anomaly detection technique for bridge strain analysis. Our approach utilizes two data sources, namely a camera installed on the bridge and a strain sensor installed underneath the bridge deck. The camera captured the motion and appearance of every passing vehicle, and the strain sensor sampled a strain response at a bridge component close to the moving axle loads. For every passing vehicle, the data from the camera and sensor were fed to two CNNs, and two feature vectors were obtained. Then, the features were compared directly in a common feature space, which was shared by the two domains of video and sensor data. The feature space was obtained by our spiral learning proposal, with an assumption of the existence of latent variables about the target vehicle, shared by the video and sensor data. The video features may play the role of an explanatory variable in the strain response.

To offset the negative influence of stormy weather on the anomaly score distribution, we also proposed the adversarial learning mechanism. The proposed adversarial network tried to detect any variables in the video features that had a direct correlation with the anomaly score. The anomaly detector learned to deceive the adversarial network and therefore prevented normal responses from being detected as anomalies. Unfortunately, we found no remarkable effect of the adversarial mechanism. Nevertheless, the adversarial mechanism suppressed the median anomaly score with small numbers of training data, potentially helping the system's deployment on real bridges with little traffic.

Our approach has two strengths. First, axle load information was not required in advance; thus, the fragile, expensive axle load meter was not required. Second, the anomaly score was obtained in a fully data-driven manner. We tested our proposals on real observation data and identified outliers of several identifiable types. In contrast, the weakness of the proposal is the interpretation difficulty of the anomaly score obtained by the neural network, which is similar to a black box.

In Chapter 7, we investigate an alternative anomaly detection approach whose output is easy to interpret by constructing a model of bridge dynamics that predicts the strain signal directly. We believe our proposals will aid bridge damage detection by identifying anomalous strain responses whose characteristics are different from those observed during the construction period.

Chapter 7

Strain Signal Prediction

Chapter 7 proposes an improved framework for anomaly detection using multimedia data obtained by multiple sensors installed on a bridge.

Bridge damage may be identified by detecting unusual mechanical behavior by the bridge components in response to passing vehicles. To detect anomalous mechanical responses, the dynamic system of the target bridge must be modeled in advance. One possible solution is to use finite element analysis (FEA) [95, 82, 55, 56]. Conventionally, dynamic simulation requires expert knowledge of mechanics, materials, and structures, in addition to accurate FEA modeling. Moreover, it requires detailed specification of the external forces applied, such as vehicle speeds, loci, and axle weights.

Our solution is to use a generative neural network, which predicts the dynamic responses of the bridge components. Fig. 7.1 shows an architecture of the proposed system, named the strain prediction system (SPS). The generative network involves two subnetworks, namely the *encoder* and the *decoder*, as introduced in the *encoder–decoder* [88, 92] approach. The encoder network collects vehicle properties in a media-fusion fashion, combining a video subnetwork and a single-sensor data subnetwork. The video is recorded by a traffic surveillance camera above the bridge and contains rich information about the target vehicle, including speed, locus (left/right position in the lane), shape, and axle positions. The sensor data are recorded by a strain sensor (or an accelerometer) underneath the bridge deck and contain information about axle weights. In this chapter, we focus on the strain responses observed at the bridge deck.

The decoder network generates the mechanical responses caused by the vehicle by taking the output of the encoder network as its input. That is, our neural network takes the raw sensor-data signal as input and outputs a decoded sensor signal, thereby modeling the transfer function between the strain sensors. To realize realistic predictions, we have improved the generative adversarial networks (GANs) [20, 71]. In Section 7.5, we demonstrate that the GAN-based approach can be applied successfully to sensor-fusion systems.

The generative network was trained by an adversarial learning algorithm improved for media-fusion analysis. Although the proposed network was not assisted by either bridge experts or FEA models, it successfully simulated strain responses for a real road bridge. Our approach has three strengths. First, axle load information was not required in advance; thus, the fragile, expensive axle load meter was not needed. Second, the bridge model was obtained in a fully data-driven manner. Finally, anomaly detection may be performed in terms of physical quantities, enabling intuitive and tangible interpretation of the detection results. Here, the weakness of the spiral learning proposed in Chapter 6 is resolved.

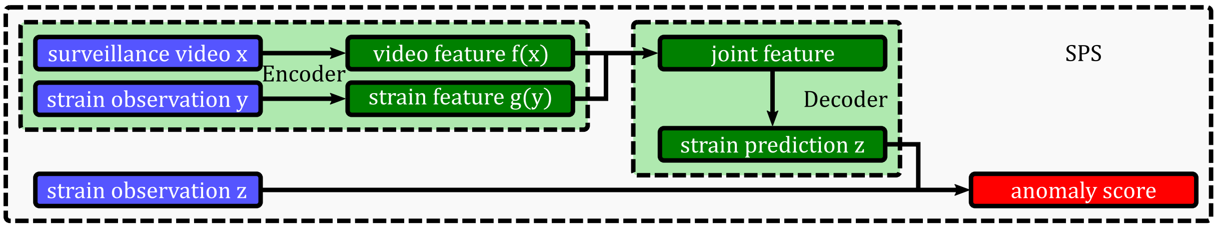


Figure 7.1: System architecture for the proposed strain prediction system.

7.1 Generator Network

Fig. 7.2 shows the architecture of the signal generator network. The network comprises two subnets, namely the encoder and the decoder. The two networks involve many preactivated residual blocks [22]. Note that we applied a *third* activation function to the output (or input) of the residual blocks in addition to the two activation functions inside the blocks. In this chapter, we used leaky ReLU [49] defined in Eq. (7.1) instead of ReLU [19] for all activation functions except for those in the output layers.

$$f(x) = \begin{cases} x & \text{if } x \geq 0, \\ 0.2x & \text{if } x \leq 0. \end{cases} \tag{7.1}$$

This may suppress the vanishing gradient in the deep neural network.

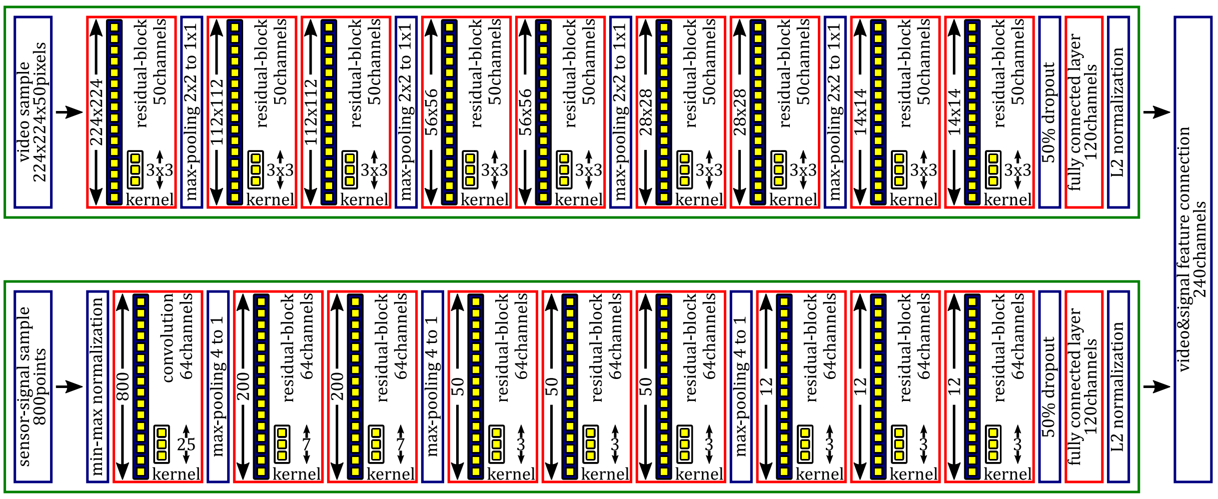
7.1.1 Encoder Network

The encoder network shown in Fig. 7.2 (a) is derived from the spiral network introduced in Chapter 6. To obtain accurate predictions, the parameters of the unknown target vehicle need to be acquired by using sensors on the target bridge. As described in Eq. (2.4), the required parameters include traveling speed, locus, axle loads, and axle positions. These parameters are obtained by the multimodal encoder, which combines two CNNs for video and strain signals.

First, the video CNN receives 50 grayscale video frames (taken over two seconds) recorded when the passing vehicle enters the camera's field of view. Each frame is resized to 224×224 pixels in advance.

Next, the strain CNN receives four-second batches of raw strain (or acceleration) signals sampled at 200 Hz. We fed the raw signal sequence directly to the CNN, following Dai et al. [13]. Each sequence starts (or ends) at the same time as the vehicle enters (LtoR) or leaves (RtoL) the bridge. Then, each sample is scaled so that its maximum and minimum values are normalized to 1 and 0, respectively, to enable effective learning.

Finally, the multimodal encoder outputs a feature vector of 240 channels by combining video


(a) Encoder network.

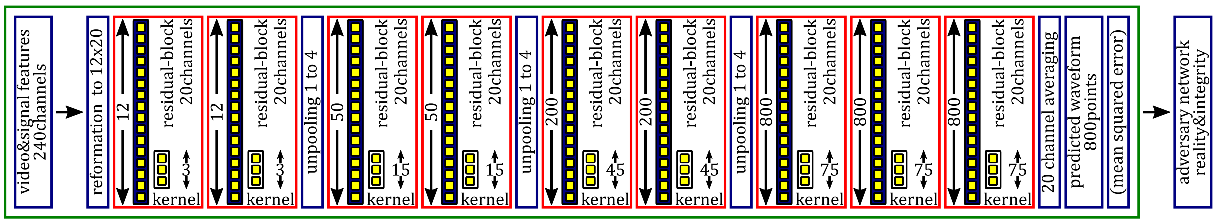

(b) Decoder network.

Figure 7.2: Media-fusion generator network architecture.

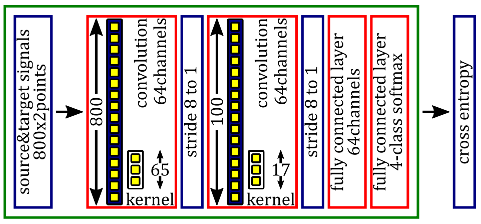


Figure 7.3: Media-fusion adversary network architecture.

and signal features of the target vehicle. To suppress overfitting, we inserted a 50% dropout between the last residual block and the following linear layer in each network. In addition, we again normalized the video and signal features individually to enable effective learning.

7.1.2 Decoder Network

The decoder network shown in Fig. 7.2 (b) receives the feature vector and predicts a normalized target waveform of 800 signal points. The decoder was designed as a deep residual network that upsamples the feature vector by three times to obtain signals at 200 Hz. The upsampling was performed simply by copying each element in the source vector into four neighboring elements in the target vector. A generous kernel width for each convolution layer was set in a manner similar to WaveGAN [14] to enable the layer to handle the low-frequency

Table 7.1: Classification problem for the adversary.

	Real $\boldsymbol{z}$	Fake $\hat{\boldsymbol{z}}$
Consistent	$\{\boldsymbol{y}_n, \boldsymbol{z}_n\}$	$\{\boldsymbol{y}_n, \hat{\boldsymbol{z}}_n \sim G(\boldsymbol{x}_n, \boldsymbol{y}_n)\}$
Inconsistent	$\{\boldsymbol{y}'_n, \boldsymbol{z}_n\}$	$\{\boldsymbol{y}'_n, \hat{\boldsymbol{z}}_n \sim G(\boldsymbol{x}_n, \boldsymbol{y}_n)\}$

strain signals seen in Fig. 5.29. The target signal was obtained by averaging 20 channels of the feature vector into a single channel in the final layer.

7.2 Adversary Network

The generator network was trained using the GAN approach [20, 71]. Typically, an adversary network examines whether the predicted data are as realistic as the observed data [20]. In addition, our proposal examines the integrity between the source signal and prediction, and the integrity should exist because they were both caused by the same target vehicle. Therefore, our adversary takes a pair comprising two signal sequences from the source sensor signal $\boldsymbol{y}$ and target sensor signal $\boldsymbol{z}$ (or prediction $\hat{\boldsymbol{z}}$), similar to pix2pix [28], and classifies the pair into four classes, as shown in Table 7.1. These four types of pairs are created uniformly for each n-th observation sample $\{\boldsymbol{x}_n, \boldsymbol{y}_n, \boldsymbol{z}_n\}$ by the sampling function s_k:

$$s_1(n) = \{\boldsymbol{y}_n, \boldsymbol{z}_n\}, \tag{7.2}$$

$$s_2(n) = \{\boldsymbol{y}'_n, \boldsymbol{z}_n\}, \tag{7.3}$$

$$s_3(n) = \{\boldsymbol{y}_n, G(\boldsymbol{x}_n, \boldsymbol{y}_n)\}, \tag{7.4}$$

$$s_4(n) = \{\boldsymbol{y}'_n, G(\boldsymbol{x}_n, \boldsymbol{y}_n)\}. \tag{7.5}$$

$\boldsymbol{y}'_n$ was selected from the samples other than $\boldsymbol{y}_n$ in the mini batch in each training iteration. Consequently, $4N$ pairs were fed to the adversary for N samples during the training.

Fig. 7.3 shows the proposed architecture of the adversary network. We modified the Model 1 CNN proposed in Chapter 5 to take the source sensor signal $\boldsymbol{y}$ and target sensor signal $\boldsymbol{z}$ as the input. The adversary network is shallower than the generator, which may suppress overfitting. The target signal input may comprise real or generated strain signals while the other input is always real but is sometimes inconsistent with the target signal.

7.3 Training Algorithm

The objective function for the adversary D is defined as the cross entropy $\mathcal{L}_D$ for the four classes and N samples:

$$\mathcal{L}_D = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{4}\sum_{k=1}^{4}\log D(k\|s_k(n)). \tag{7.6}$$

The function D takes a pair created by s_k selected uniformly as input and estimates the probability that the input pair is classified to the k-th class in Table 7.1.

The generator G was trained so the prediction $\hat{\boldsymbol{z}}$ looks real and consistent with the observation $\boldsymbol{y}$. To stabilize the GAN optimization, we added an auxiliary MSE element to the cross entropy for G, following Yang et al. [97] to obtain Eq. (7.7).

$$\mathcal{L}_G = \frac{1}{N}\sum_{n=1}^{N}\left\{\log D(1\|s_3(n)) + \lambda\|\hat{\boldsymbol{z}} - \boldsymbol{z}\|^2\right\}. \tag{7.7}$$

λ was set to 1. Both networks were trained in a mini-batch fashion so that the losses $\mathcal{L}_G$ and $\mathcal{L}_D$ were minimized. In each batch iteration, G was updated first, then D was updated.

7.4 Training and Evaluation Data

In Section 7.5, we demonstrate our proposal for the case of Bridge C. As shown in Fig. 4.1 (a), we installed a traffic surveillance camera and some strain sensors on the bridge. The camera was installed at the bridge entrance to capture images of vehicles in close proximity to the vehicle bodies. At the same time, we deployed four highly sensitive strain sensors underneath the bridge deck to collect strain responses in the direction orthogonal to the bridge axis. We also installed an accelerometer at the center of the target span. The sensors are shown as the four red and yellow triangles denoted S1P4, S2P4, S3P4, S4P4, and A3ZP8, respectively, in Fig. 4.1. All of the sensors were synchronized except for the camera, and their sampling rate was 200 Hz. The frame rate of the camera was set to 25 frames per second. As described in Chapter 5, the response signals themselves contain rich information about the vehicles that may include speed, loci, and axle numbers, although the accuracy may be somewhat lower than that derived from the video data. However, axle loads may be better obtained from strain (or acceleration) data than from video data.

In preparation for the experiments, we needed the ground-truth data of target vehicles on Bridge C. We utilized the dataset DS601 prepared in Chapter 4 for this purpose. DS601 was created via a traffic surveillance system (TSS) [32] based on the Faster R-CNN [73] and contains information about 996,093 vehicles that crossed the bridge between 08:00 and 16:00 from November 5, 2016, to April 28, 2017. In this work, we ignored all vehicles with two axles because civil engineers are mainly interested in heavy vehicles.

We then collected videos of the vehicles and sensor signals caused by the vehicles. The n-th record is described as a triplet $\{\boldsymbol{x}_n, \boldsymbol{y}_n, \boldsymbol{z}_n\}$, which is composed of a video $\boldsymbol{x}$, source sensor signal $\boldsymbol{y}$, and target signal $\boldsymbol{z}$ for the n-th vehicle. The dataset was divided randomly into two subsets. The first subset was treated as *trainval* data, and the second was used for *evaluation*: 80% of the *trainval* data were assigned as *training* data, with the remaining 20% being *validation* data, which were utilized for early stopping [98]. The evaluation was performed using a model that updated the minimum of the MSE between the observed waveforms and the predictions. The trainval dataset therefore involved 17,757 LtoR and 20,996 RtoL vehicles, and the evaluation dataset involved 17,967 LtoR and 21,078 RtoL vehicles.

7.5 Experimental Results

We implemented two derivative generator models on Chainer [68] 5.0.0. One was the GAN model described in Section 7.3, and the other was an MSE model trained without the GAN mechanism. They were accelerated by a GPU (NVIDIA GeForce GTX 1080 Ti) utilizing CUDA [60] 9.2. We also employed the AMSGrad [72] for optimization, and the mini-batch size was set to 10. The two models were trained over 200 epochs, and the evaluation processes were performed using the early stopping approach [98]. In addition, we trained the plain spiral network described in Chapter 6 and compared anomaly scores for the real observation data and waveforms predicted by the two generator models.

7.5.1 Strain Prediction Appropriateness

First, we evaluated the signal correlation between the observation $\boldsymbol{z}$ and the prediction $\hat{\boldsymbol{z}}$. Unfortunately, there are no widely accepted metrics for waveform similarity beyond squared distance and cross-correlation. In this work, we focused on the fact that the values of signal points could be estimated by a linear combination of influence lines and therefore evaluated the correlation between the signal points of the ground truth and the prediction. It should be noted that the distributions of the signal values were not uniform but biased strongly toward 0 μST, as shown in Fig. 4.2. Therefore, we used the rank correlation coefficient τ's [36] as metrics. Table 7.2 shows the average τ coefficients and the MSEs for the evaluation data. From these results, the proposed GAN model was able to achieve a strong correlation between observation and prediction.

Table 7.2: Kendall τ coefficients and MSEs for the GAN.

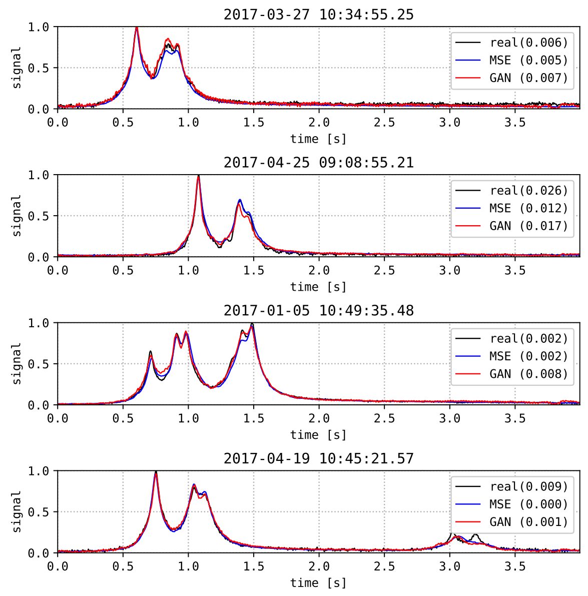
LtoR				RtoL			
Source	Target	$\overline{\tau}$	$\overline{\|\hat{\boldsymbol{z}} - \boldsymbol{z}\|_2^2}$	Source	Target	$\overline{\tau}$	$\overline{\|\hat{\boldsymbol{z}} - \boldsymbol{z}\|_2^2}$
S3P4	S1P4	0.748	1.242	S1P4	S3P4	0.747	2.756
	S2P4	0.687	2.018		S4P4	0.730	3.035
S4P4	S1P4	0.691	1.268	S2P4	S3P4	0.756	2.645
	S2P4	0.674	1.881		S4P4	0.720	3.015
A3ZP8	S1P4	0.676	4.559	A3ZP8	S3P4	0.690	6.906
	S2P4	0.641	6.150		S4P4	0.693	7.932

7.5.2 Strain Prediction from Strain Data

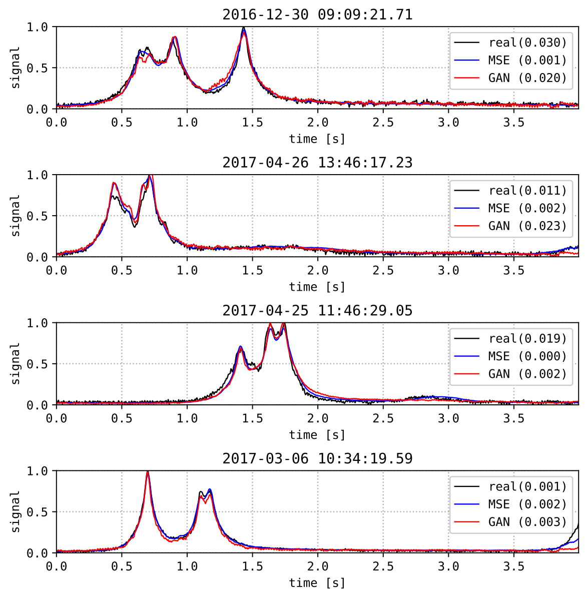
After the waveform prediction for LtoR vehicles in the evaluation dataset, we sorted the waveforms in terms of squared error and extracted the first and third quantiles. As described in Section 6.3.2, the traffic data contained some false large vehicles; quantile extraction may remove such noise components. In addition, we could grasp the whole tendency of prediction fitness by focusing on both quantiles.

First, we predicted waveforms for LtoR vehicles that should be observed by the strain sensor S1P4. As shown in Fig. 4.1 (a), the sensor was installed to be under the wheels of most LtoR vehicles. Fig. 7.4 and Fig. 7.5 show examples of predictions using strain sensors S3P4 and S4P4, respectively. These sensors were installed underneath the opposite lane and may be less sensitive to the passing axle loads than S1P4. The numbers in brackets indicate the anomaly scores for real and fake signals. For both quantiles, the proposed network generated realistic fake signals that were indistinguishable from the real observation data. As seen in the graphs, the waveforms generated by the MSE model were much smoother than those obtained by the GAN model. Consequently, the GAN simulated the macroscopic signals and the noise components. The strain peaks indicate the times at which axles passed over the sensors, and the two generators successfully simulated these peaks both in terms of peak heights and times.

Second, we predicted waveforms for LtoR vehicles that should be observed by the strain sensor S2P4. As shown in Fig. 4.1 (a), the sensor was installed nearby the road center in the LtoR lane. Fig. 7.6 and Fig. 7.7 show examples of predictions using strain sensors S3P4 and S4P4, respectively. These sensors were installed underneath the opposite lane and may be less

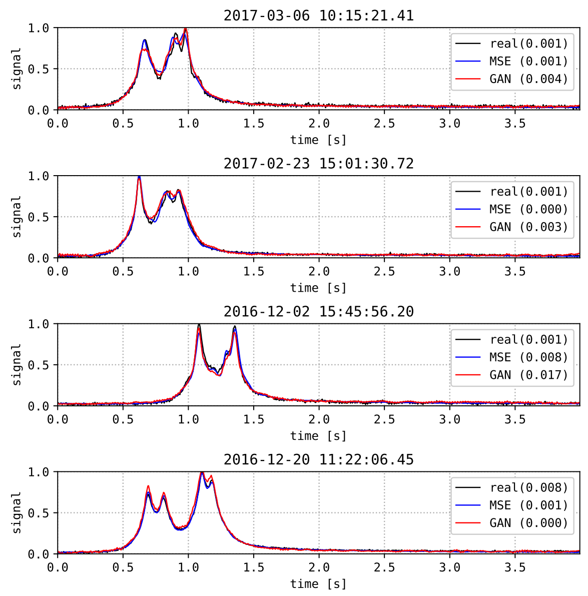


(a) 1st quartile.

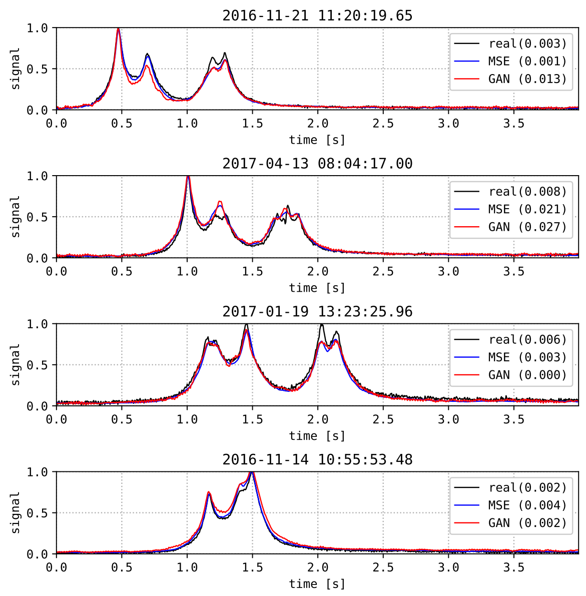


(b) 3rd quartile.

Figure 7.4: Strain prediction from S3P4 to S1P4 for LtoR vehicles.



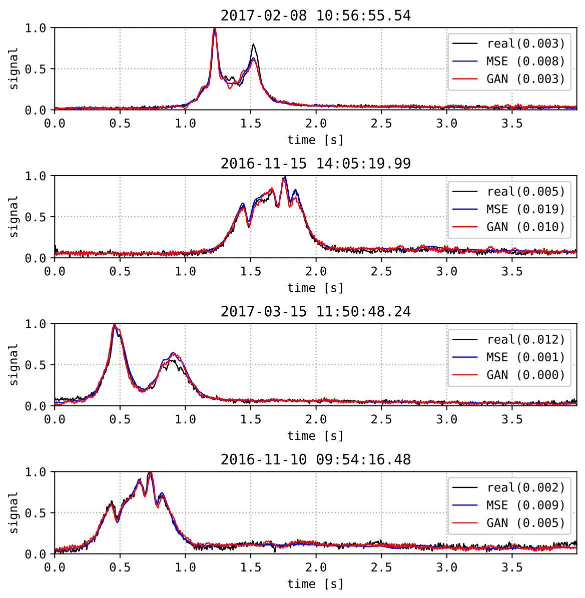
(a) 1st quartile.



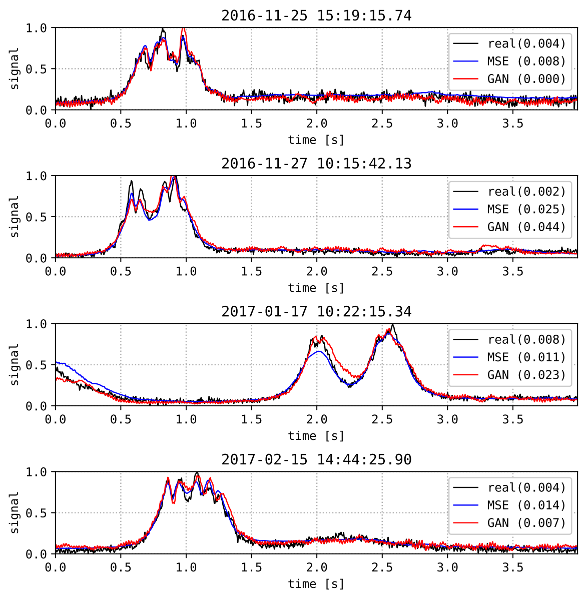
(b) 3rd quartile.

Figure 7.5: Strain prediction from S4P4 to S1P4 for LtoR vehicles.

sensitive to the passing axle loads than S2P4. The numbers in brackets indicate the anomaly scores for real and fake signals. For both quantiles, the proposed network generated realistic signals that were indistinguishable from the real observation data. The sensors S1P4 and S2P4 tend to sample somewhat different waveforms, as shown in Fig. 4.2. By focusing on the peaks and skirts of the waves, we found that the skirt heights relative to the peaks were much greater

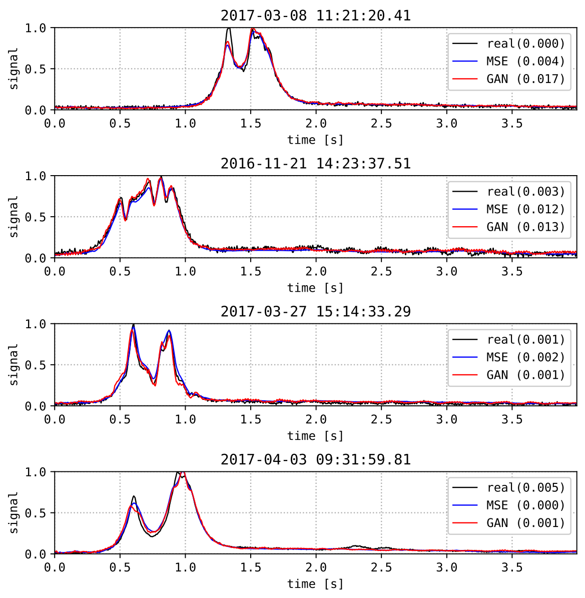


(a) 1st quartile.

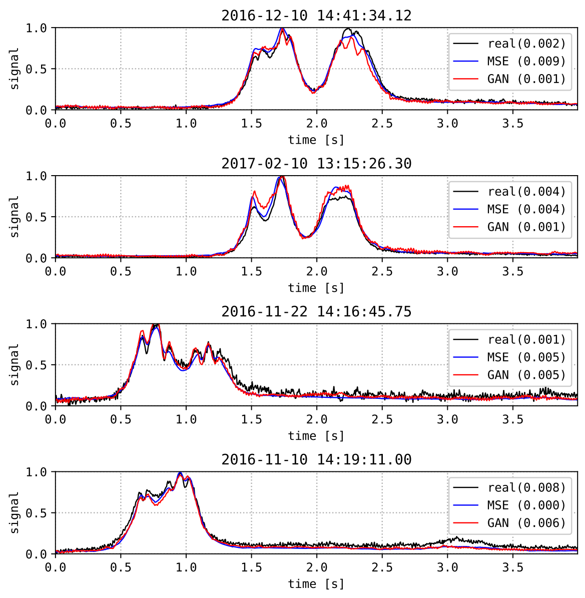


(b) 3rd quartile.

Figure 7.6: Strain prediction from S3P4 to S2P4 for LtoR vehicles.



(a) 1st quartile.



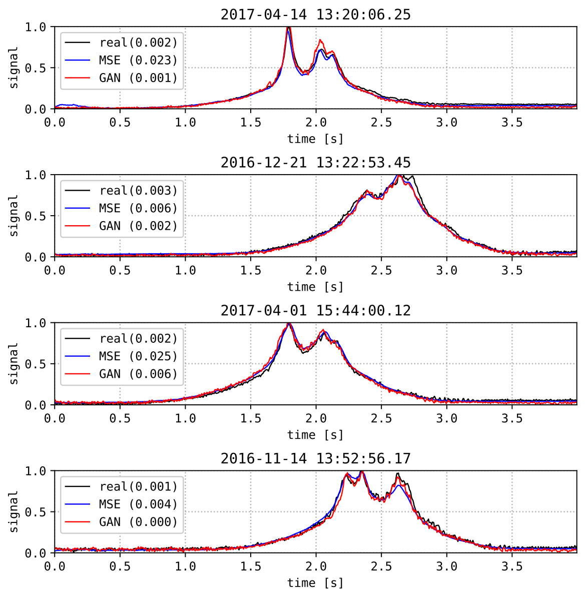
(b) 3rd quartile.

Figure 7.7: Strain prediction from S4P4 to S2P4 for LtoR vehicles.

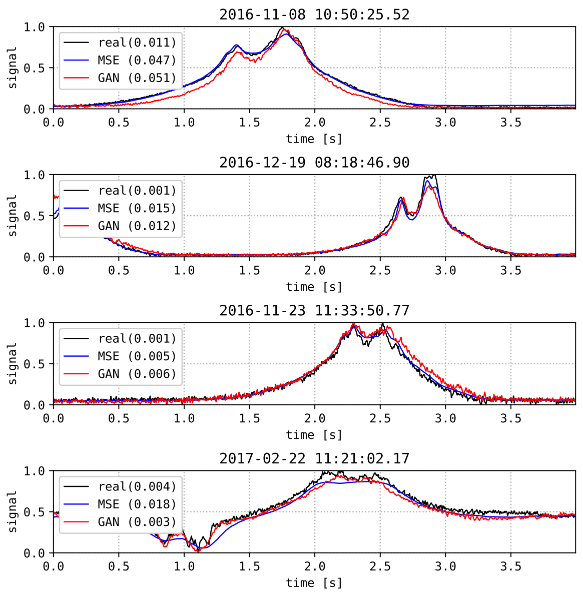
than those observed by S1P4. This tendency can be seen in the prediction results.

As with LtoR vehicles, we trained the GAN model and MSE model for the RtoL vehicles and extracted the first and third quantiles in terms of squared error.

First, we predicted waveforms for RtoL vehicles that should be observed by the strain sensor S3P4. As shown in Fig. 4.1 (a), the sensor was installed near the center of the road in the

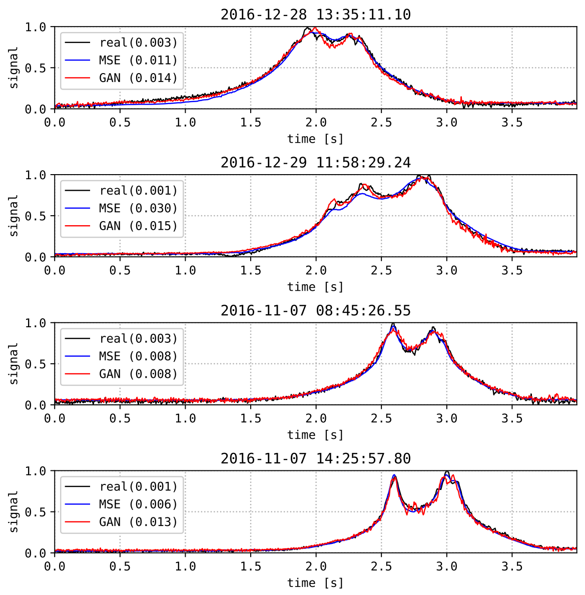


(a) 1st quartile.

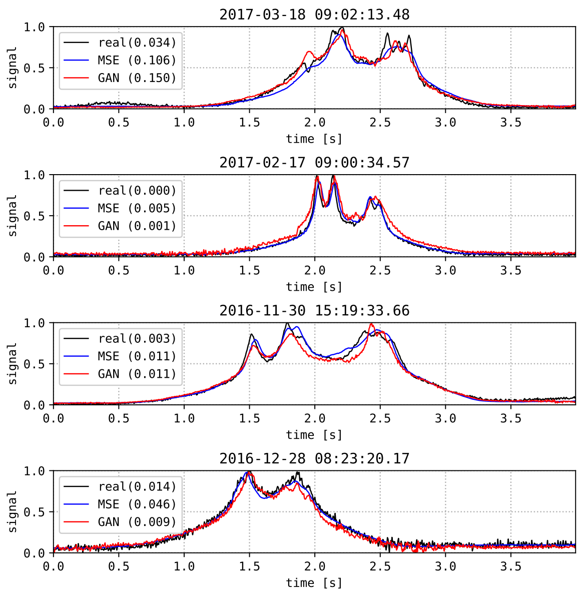


(b) 3rd quartile.

Figure 7.8: Strain prediction from S1P4 to S3P4 for RtoL vehicles.



(a) 1st quartile.



(b) 3rd quartile.

Figure 7.9: Strain prediction from S2P4 to S3P4 for RtoL vehicles.

RtoL lane. Fig. 7.8 and Fig. 7.9 show examples of predictions using strain sensors S1P4 and S2P4, respectively. These sensors were installed underneath the opposite lane and may be less sensitive to the passing axle loads than S3P4. The numbers in brackets indicate the anomaly scores for real and fake signals. For both quantiles, the proposed network generated realistic signals that were indistinguishable from the real observation data. Because of the asymmetric structure of the box girder, the shapes of the observed signals were very different from those in Fig. 7.4. These differences can be seen best in the skirts of the waves, whose heights relative to the peaks were much greater. Again, our proposal successfully simulated these gentle slopes that start and end just as the vehicle enters and leaves the target span on the bridge. In contrast, spikes in the predicted waveforms were flatter than those in the previous graphs because of the lack of clarify of the strain peaks in the observation data.

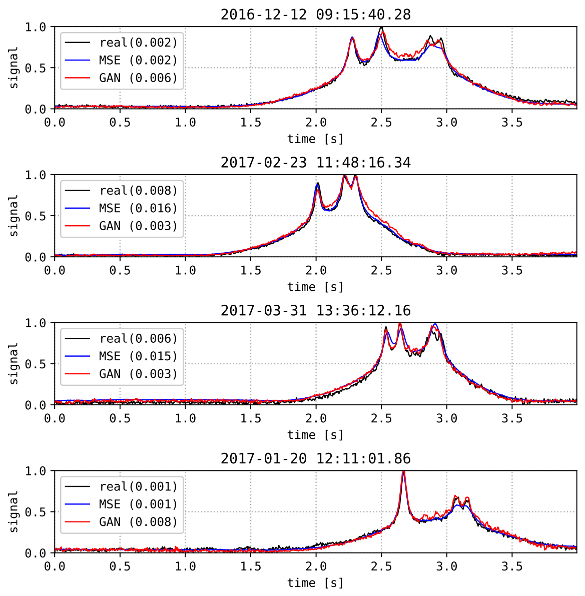
Lastly, we predicted waveforms for RtoL vehicles that should be observed by the strain sensor S4P4. As shown in Fig. 4.1 (a), the sensor was installed to be under the wheels of most RtoL vehicles. Fig. 7.10 and Fig. 7.11 show examples of predictions using strain sensors S1P4 and S2P4, respectively. These sensors were installed underneath the opposite lane and may be less sensitive to the passing axle loads than S4P4. The numbers in brackets indicate the anomaly scores for real and fake signals. The observed waveforms had a tendency similar to combining the shape tendencies of the S2P4 signals for LtoR vehicles and S3P4 signal for RtoL vehicles. In other words, the skirt heights relative to the peaks were great but did not hide the spikes completely. Again, our proposal successfully simulated both these gentle slopes and spikes.

7.5.3 Strain Prediction from Acceleration Data

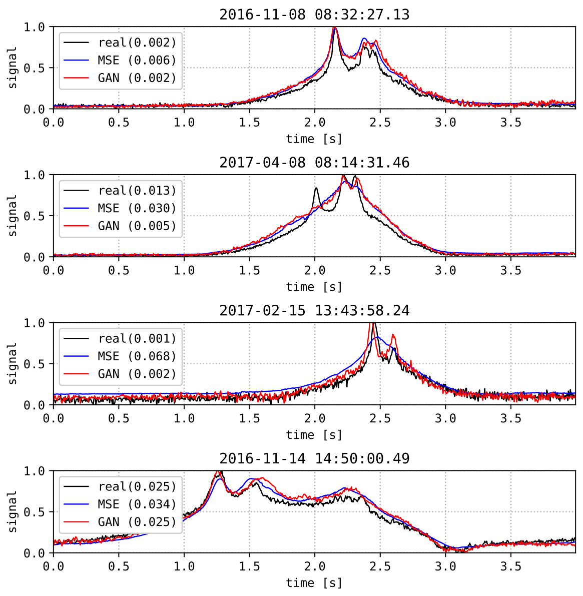
Section 7.5.3 explores the probability of using an accelerometer as the source sensor instead of using strain sensors, as explored in Section 7.5.2. As seen in Fig. 4.3, the acceleration at the center of the bridge span had poor spatial locality and tended to react to distant vehicles, possibly making acceleration analysis more difficult than strain analysis. We expected that this difficulty must have had a great influence on the accuracy of the axle load estimation and therefore may lead to incorrect waveform prediction. The experiments were performed in the same manner as Section 7.5.2.

First, we predicted waveforms for LtoR vehicles, which should be observed by the strain sensors S1P4 and S2P4. As shown in Fig. 4.1 (a), the sensors were installed to be under the wheels of LtoR vehicles. Fig. 7.12 and Fig. 7.13 show examples of predictions for the two sensors using accelerometer A3ZP8. A3ZP8 was installed at the center of the target span and was far from the target strain sensors. The numbers in brackets indicate the anomaly scores for real and fake signals. Surprisingly, the proposed network generated realistic signals for the first quantiles with low anomaly scores. However, the residual errors were much greater than those in Section 7.5.2. For the third quantiles, the strain peaks were shifted a little and their heights were incorrect. Thus, the generator failed to locate and measure moving axle loads.

Next, we predicted waveforms for RtoL vehicles that should be observed by the strain sensors S3P4 and S4P4. As shown in Fig. 4.1 (a), the sensors were installed to be under the wheels of RtoL vehicles. Fig. 7.14 and Fig. 7.15 show examples of predictions for the two sensors using

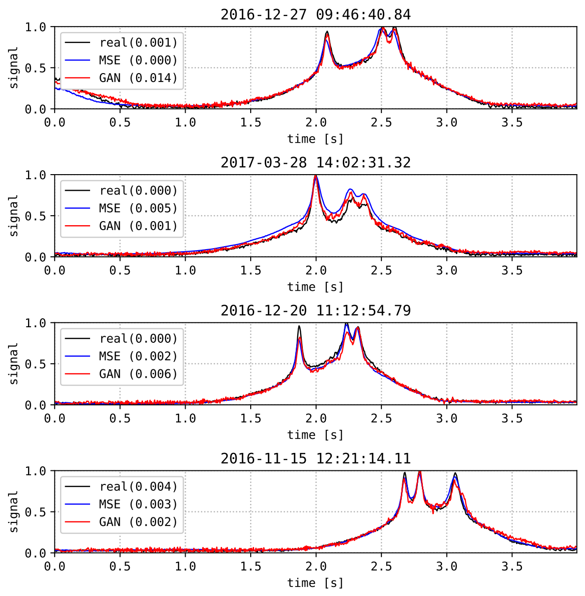


(a) 1st quartile.

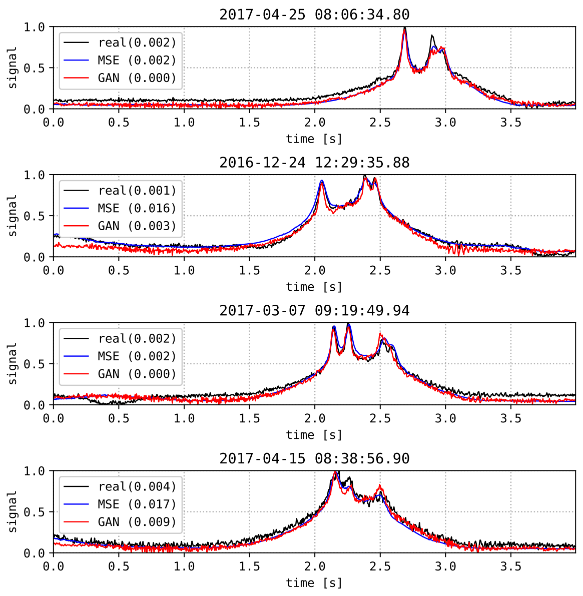


(b) 3rd quartile.

Figure 7.10: Strain prediction from S1P4 to S4P4 for RtoL vehicles.



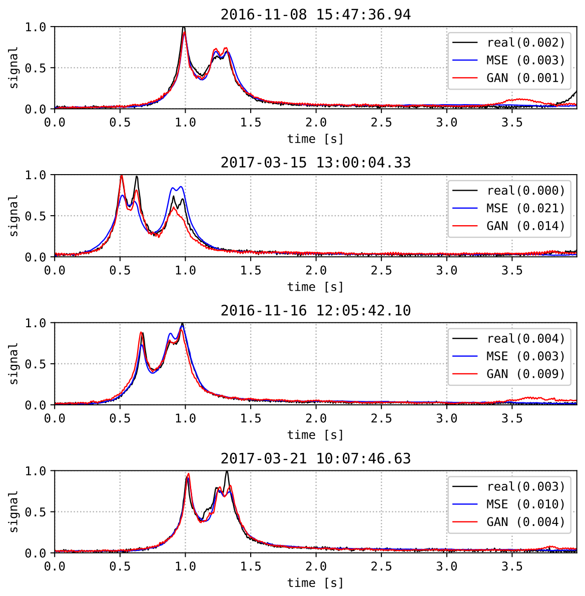
(a) 1st quartile.



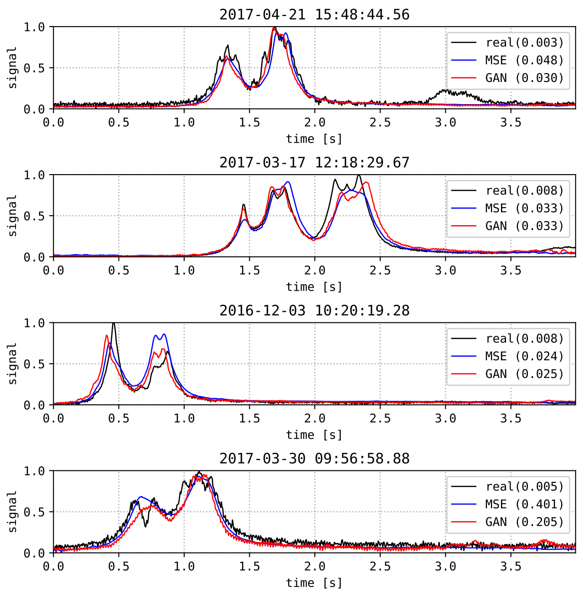
(b) 3rd quartile.

Figure 7.11: Strain prediction from S2P4 to S4P4 for RtoL vehicles.

accelerometer A3ZP8. The numbers in brackets indicate the anomaly scores for real and fake signals. As with LtoR vehicles, the proposed network generated realistic signals for the first quantiles, with low anomaly scores. Our proposal successfully simulated these gentle slopes that start and end just as the vehicle enters and leaves the target span on the bridge. In contrast, for the third quantiles, the strain peaks were shifted a little, and their heights were incorrect. In

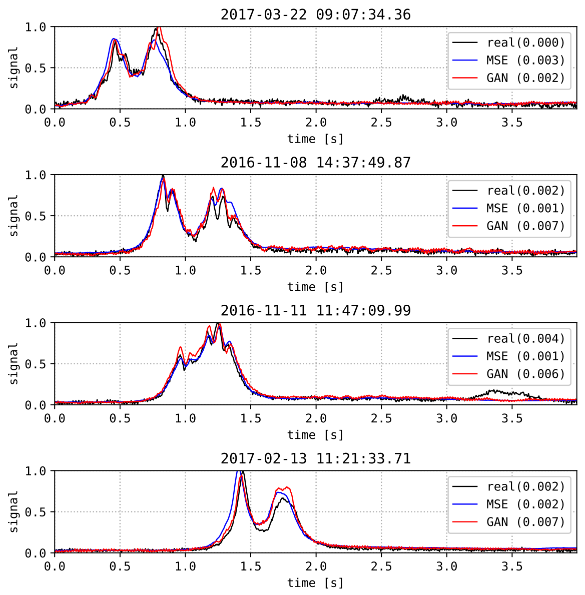


(a) 1st quartile.

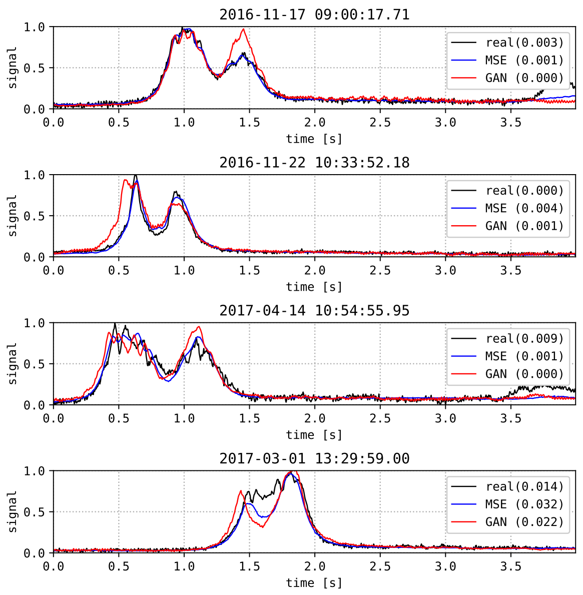


(b) 3rd quartile.

Figure 7.12: Strain prediction from A3ZP8 to S1P4 for LtoR vehicles.



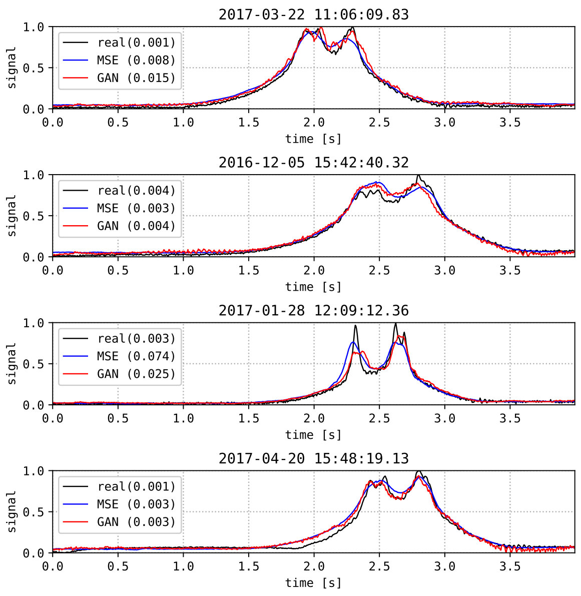
(a) 1st quartile.



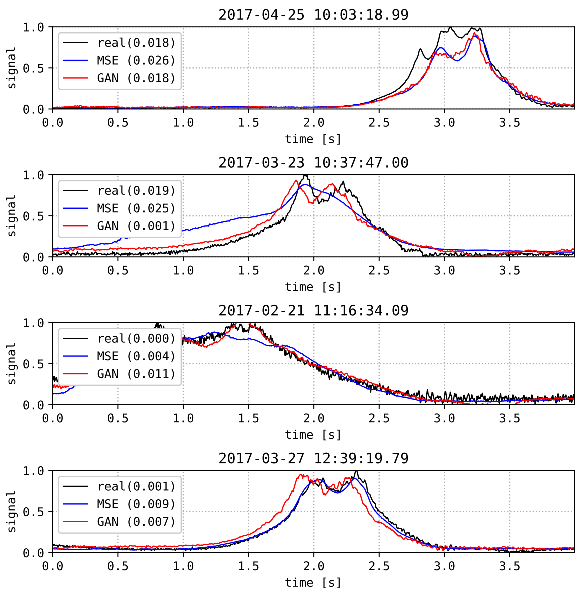
(b) 3rd quartile.

Figure 7.13: Strain prediction from A3ZP8 to S2P4 for LtoR vehicles.

conclusion, the modeling of bridge dynamics using an accelerometer can be greatly improved.

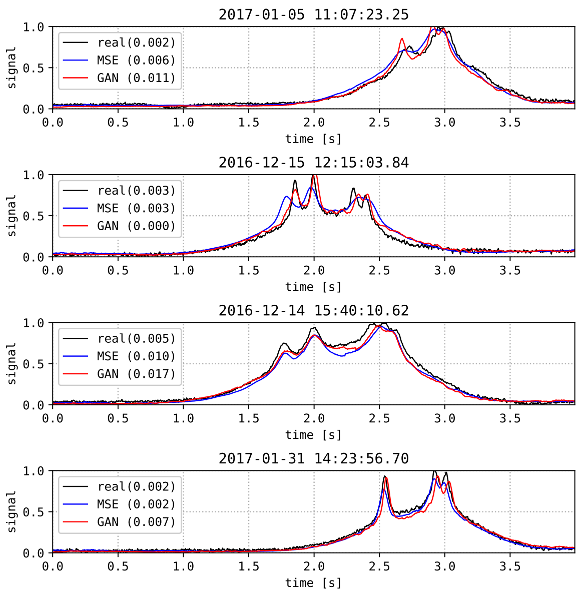


(a) 1st quartile.

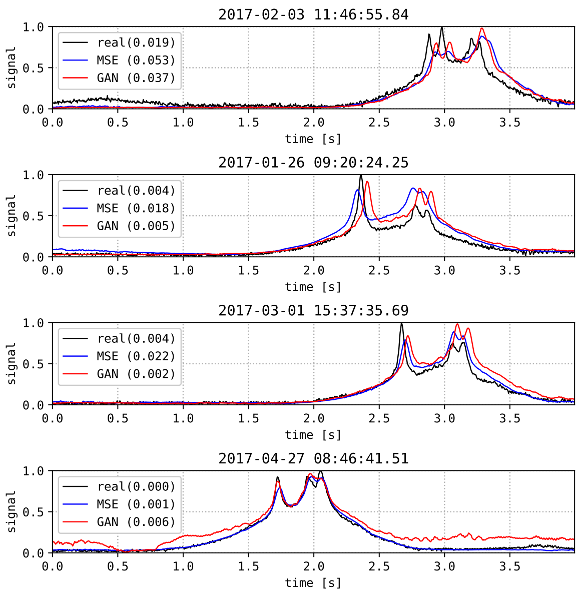


(b) 3rd quartile.

Figure 7.14: Strain prediction from A3ZP8 to S3P4 for RtoL vehicles.



(a) 1st quartile.

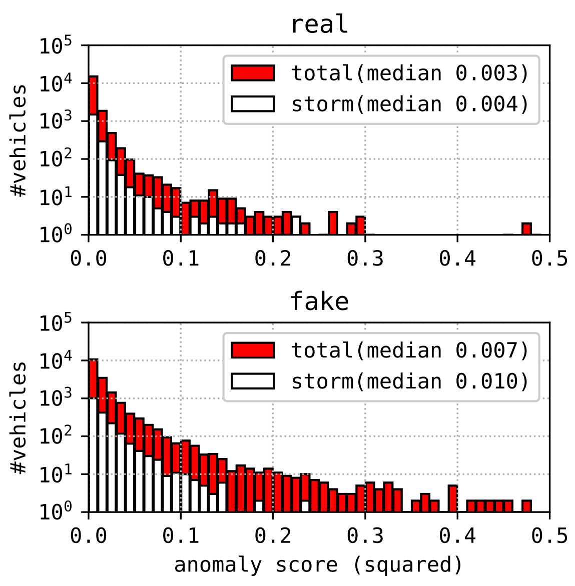


(b) 3rd quartile.

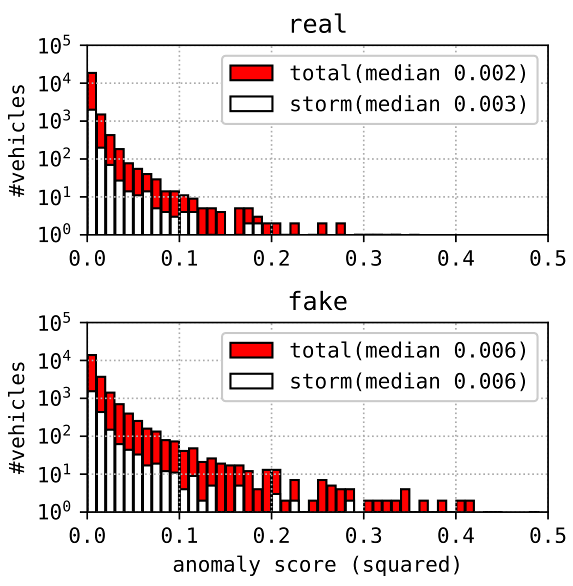
Figure 7.15: Strain prediction from A3ZP8 to S4P4 for RtoL vehicles.

7.5.4 Anomaly Score Distribution

We evaluated the distribution of anomaly scores in a similar manner to Section 6.3.1. We used a SpiNet which was trained by the plain spiral learning algorithm with pairs of real observations

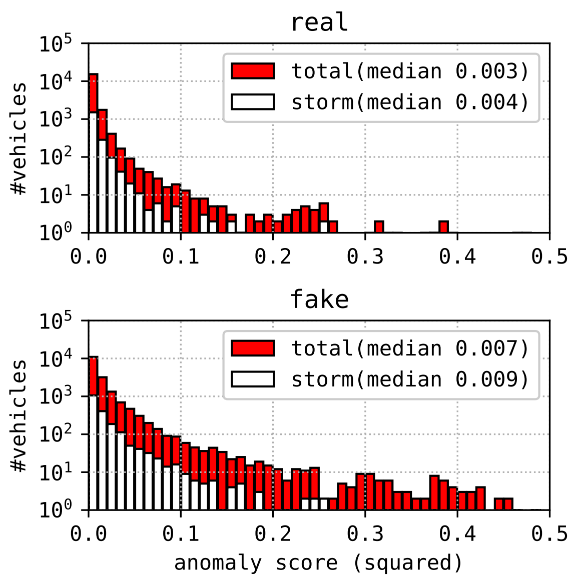


(a) LtoR (from S4P1 to S1P4).

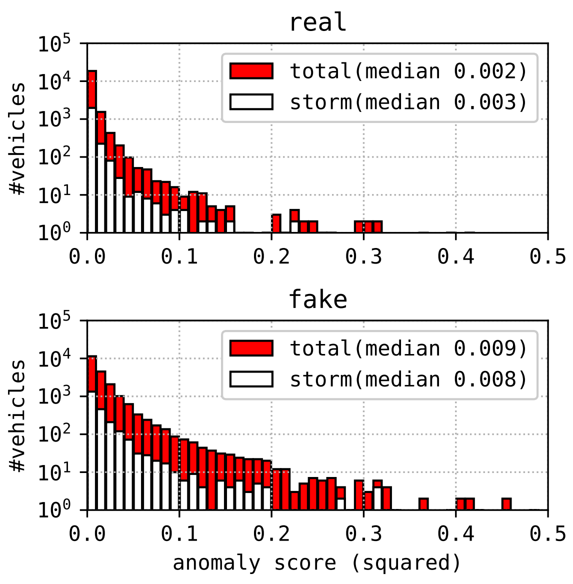


(b) RtoL (from S1P1 to S4P4).

Figure 7.16: Comparison of anomaly score distributions when using strain sensors.



(a) LtoR (from A3ZP8 to S1P4).



(b) RtoL (from A3ZP8 to S4P4).

Figure 7.17: Comparison of anomaly score distributions when using an accelerometer.

for the video and target sensor data. We compared two cases, namely real observations and prediction by the GAN model. To validate the robustness against weather conditions, we also drew histograms for stormy days with heavy snow or haze for several hours. Table 6.1 lists the stormy days.

Fig. 7.16 shows the logarithmic histograms of the anomaly scores estimated for real observations and fake waveforms predicted by the GAN model. The two pairs of the upper graphs and lower graphs were obtained by using the two spiral networks that were trained for LtoR vehicles and RtoL vehicles, respectively. Therefore, we could compare the upper and lower graphs directly for the LtoR and RtoL cases. As seen in Fig. 7.16, we found a remarkable difference between the real and fake signals, and the median score for the fake signals was much higher than those for the real observations. The difference is a mystery because the fake signals were judged as anomalous by using video data, even though the signals were obtained using the same videos. One possible factor was the source signal $\boldsymbol{y}$, which was not considered in the calculation of the anomaly scores.

Fig. 7.17 shows the logarithmic histograms of the anomaly scores when using an accelerometer as a source for the prediction. The distributions indicate similar tendencies to the cases of using strain sensors. Thus, the predicted waveforms shown in Section 7.5.3 were not abnormal as long as they were compared with the video data in the common feature space. In contrast, their MSEs were much greater than the MSEs using strain sensors. Consequently, the waveforms predicted when using accelerometer A3ZP8 were sufficiently natural but did not reflect accurate axle loads. In conclusion, the technique of extracting axle loads from acceleration signals needs to be more sophisticated.

7.6 Discussion

In this work, we focused on strain responses as the target sensor data, although the bridge components may produce many kinds of mechanical responses when a vehicle crosses the bridge. Strain responses on the bridge deck have strong spatial and temporal locality, with the strain meters reacting very little to axle loads at distant points and recovering to the bias points rapidly after a vehicle departs. In contrast, other sensors, such as an accelerometer, at the center of the bridge span can react to distant vehicles, and vibrations at the natural frequencies are likely to be persistent. These properties make vibration analysis more difficult compared with strain analysis. Whenever anomalous strain responses are found by comparing the predicted and observed responses, we may locate the area of damage by reference to axle positions, as has been proposed in some studies [10, 27]. However, strain sensors are prone to peeling off the surface of bridge components and require regular inspection. This is one motive for using accelerometers instead of strain sensors in civil engineering applications [81]. Therefore, we should develop a generator for vibration response prediction as an alternative to strain response prediction. Note that the proposed generator may aid in the inspection process for strain sensors themselves.

In Section 7.5.3, we demonstrated strain predictability by using accelerometer A3ZP8 for the source sensor data. Unfortunately, the performance was lower than using strain sensors as sources; thus, the accelerometer could not replace the strain sensors completely. The graphs explored in Section 7.5.3 suggested that the encoder network failed to locate and measure moving axle loads. The failure may be explainable by the fact that the accelerometer had poor sensitivity and spatiotemporal locality to the individual axles, as shown in Fig. 4.3. To improve predictability, we should change the installation position to be much closer to the road surface, as discussed in Chapter 5.

It should be noted that our generator did not take a random variable as input, unlike the previous GAN studies [20, 71]. This was because we required deterministic bridge models, and the contribution of the random input was not obvious in our study. However, we may explore a technique of handling probabilistic behavior of the bridge by exploiting random variables.

7.7 Conclusion

In this chapter, we have proposed a novel media-fusion framework for detecting anomalous strain responses caused by vehicles. The proposed GAN enables direct translation between strain sensors installed underneath the bridge deck by consulting a traffic surveillance camera on the bridge. The video features may specify vehicle properties including speed, locus, and axle positions as explanatory variables in the prediction. Although the video may lack information about axle loads, this can be compensated for by analyzing the source sensor signals. We tested our proposals on real observation data with the results demonstrating highly accurate predictions of measured waveforms.

We expect that bridge damage and sensor faults may be revealed by comparing the error distributions of predictions collected soon after the bridge's construction with the current ones. Compared with the direct comparison of video and strain data in a common feature space, the proposal in Chapter 7 enables visualization of changes in physical quantities. We also aim to investigate sensor types other than strain meters for use as alternative signal sources.

Chapter 8

Conclusion

In this dissertation, the analysis of multimedia data including traffic surveillance cameras, strain sensors and accelerometers on bridges is explored. The fundamental interest of this study is the application of data analysis techniques to the realization of social CPSs, acquiring expert knowledge in a completely data-driven fashion. The application of a deep CNN to multimedia sequential data is suited to this field. Our approach was oriented to extracting information richness from each sensor data, which could be exploited widely in the application of SHM. This approach was consistently applied to the three problems that are vehicle detection, anomalous signal detection, and signal prediction, and was improved progressively from a single medium to media fusion addressing all three problems. The study has shown that the bridge dynamics can be represented by a multimedia CNN, and extraction and utilization of traffic situations on the bridge have been demonstrated.

In this chapter, we provide the general conclusion of this dissertation and review the previous chapters from the global perspective. Finally, we state the future prospects in this research field.

8.1 Review

Chapter 1 introduced this dissertation. Conventional decision systems have been developed individually in their closed application fields. Backed by technical improvements, the traditional form of decision systems is forced to evolve into social CPSs. A CPS integrates the specialized decision systems by organizing and sharing resources of sensing, communication, situation analysis, and actuation, which work together via the Internet. A CPS is responsible for all domains of social activities and enables maintenance of social infrastructures in depopulated areas. The main target of this dissertation is bridge maintenance. To realize fully automated bridge health assessment, a long-term monitoring system involving heterogeneous, inexpensive sensors is deployed. The aim of this work is to determine whether small signs of bridge deterioration can be detected by applying data mining techniques to long-term sequential data. The monitoring system provides multiple interfaces for different types of experts, including bridge owners, maintainers, civil engineers, and data scientists, who may collaborate with each other on the system. The fundamental interest of this study is in applying deep CNN techniques to the problem of bridge damage detection, which enables flexible application to any bridges in the real world. The study's challenge is composed of two steps: extracting information richness from the individual sensor data and exploiting the richness for modeling bridge dynamics. Because there has been no collection of large-scale data for machine learning related to bridge damage, the development of an anomaly detection approach is required.

Chapter 2 introduced existing studies of structural health monitoring. Although the true mechanism for damage progression on real bridges is yet to be fully clarified, there are many approaches to bridge health assessment. The approaches can be divided into destructive and nondestructive inspection approaches, and the latter must be developed to realize inexpensive, frequent inspection. Traditional nondestructive techniques, including visual and hammering inspection, can hardly identify damage inside the bridge components, and radiographic inspection is costly and difficult to automate. Sensor-fusion techniques can be an alternative, and dynamic analysis may be a definitive approach. To exploit this approach, a bridge model must be estimated in advance. Conventionally, such a model was obtained via FEA modeling, but FEA requires specialized knowledge about mechanics, materials, and structures. Therefore, implicit modeling techniques have been developed, including natural oscillation models and influence line models. In addition, BWIM systems were utilized for estimation of an accumulated volume of heavy vehicles on a bridge. BWIM exploits the influence line model and requires additional sensors for vehicle detection, and they are typically installed underneath the bridge deck.

Chapter 3 reviews image processing and mining techniques for video data. The CNN is a derivative of an artificial neural network that can decrease the number of network parameters drastically and thus has achieved superior scores for image recognition tasks. The GAN is a framework for training a generative neural network, where a generative model creates a random fake image by accepting a random variable as input and an adversary model examines the authenticity of the fake image. After iterative training of the two models, the generative model generated realistic images that were indistinguishable from the real images. To detect a vehicle, several approaches such as background subtraction, object detection, and semantic segmentation may be a solution. Background subtraction is a basic technique for extracting pixels for moving objects. Object detection utilizes features extracted from video frames and recently, CNNs have achieved formidable performance. Semantic segmentation extracts object shapes instead of bounding boxes, considering features extracted by a deep CNN. Some of the introduced techniques are utilized in later chapters.

Chapter 4 presented a video analysis framework for preparing a traffic dataset to be utilized in experiments in later chapters. The proposed system, the TSS, combines background subtraction and CNN for object detection to detect every passing vehicle with moderate accuracy and throughput. The TSS was first proposed for the purpose of damped vibration analysis for real bridges, and the chapter also presents the vibration analysis system which utilizes the TSS. The TSS was evaluated on two real bridges, and robustness against bad weather conditions was evaluated. We then improved the weatherability of the TSS and created traffic datasets for the two bridges, named DS601 and DS801. The datasets contained vehicle properties including traveling lane, speed, locus, and number of axles for every passing vehicle on the bridges. Some vehicles selected from DS601 at random were compared with ground-truth data created by hand, and the dataset accuracy was demonstrated.

Chapter 5 presented a vehicle detection system for a BWIM system that used only a single strain (or acceleration) sensor. The deep sensing proposal exploits information richness of individual sensor data and successfully extracted vehicle properties using sensors underneath the bridge deck. This proposal may extend the role of each individual sensor and may simplify the whole sensing system deployed on bridges. In this chapter, three derivative CNNs were proposed that learn four tasks, namely vehicle detection, speed estimation, locus estimation, and axle counting, taking a multitask learning approach. The proposed CNNs were trained using the traffic datasets prepared in Chapter 4 until validation losses converged. Then, the model generalizability was estimated by fivefold cross-validation, which divides the whole dataset into three subsets for training, validation, and evaluation at random. For both bridges, the CNNs successfully extracted vehicle properties, but the performance for the steel bridge could be improved. The high detectability of speed and locus indicates the possibility that the CNNs learned the influence surface for the two-dimensional road surface. Finally, we implemented and demonstrated a BWIM system using only a single strain sensor.

Chapter 6 presented an anomaly detection system that exploits the deep sensing approach. The hypothesis that the deep sensing CNNs learned the response characteristics for moving axle loads led to the possibility of bridge damage detection in terms of the response function obtained by the deep sensing approach. Such a response function may accept a passing vehicle as an explanatory variable, and vehicle properties excluding axle loads are obtainable using a surveillance camera. Because collecting axle loads is difficult on real bridges, we proposed an anomaly detection system that extracts vehicle properties excluding axle loads from video and sensor data and compares them in a common feature space. Bridge damage may be detectable by collecting accumulated inconsistencies between video and sensor data in the common feature space. The feature comparison was implemented in a multimodal CNN which combines two subnetworks for video and sensor data. The subnetworks were trained using DS601 as for the deep sensing approach in Chapter 5. The experimental results show the possible effectiveness of the proposed spiral learning approach for anomalous strain detection, and four types of anomaly were discovered in a case study. In addition, an additional mechanism of adversarial learning was proposed to eliminate the effect of weather conditions on anomaly detection. Although we could not confirm the effectiveness of the mechanism, the adversarial learning mechanism suppressed anomaly scores with small training data. The problem of this work was the difficulty regarding interpretation of the estimated anomaly score.

Chapter 7 presented a modeling technique for bridge dynamics utilizing a surveillance camera and two strain sensors. The proposal aims to detect anomalous strain responses for Chapter 6, but anomaly detection is based on physical quantities that are easy to interpret. The dynamic model was implemented by a multimodal generative CNN combining encoder and decoder

CNNs. The encoder accepts a pair of video and sensor data as inputs and is expected to extract vehicle properties such as traveling speed, locus, shapes, and axle loads. The decoder generates strain waveforms that should be observed by the other sensor by taking the vehicle features as input. To obtain realistic waveform prediction, we also implemented an adversarial CNN that examines the consistency between the source sensor signals and prediction. The generator was then trained using a multitask learning approach in which the mean squared error of prediction was minimized while the adversarial loss was maximized. The experimental results show the successful acquisition of bridge dynamic models that may enable early warning of bridge deterioration in a fully data-driven manner in the future.

8.2 Future Prospects

Social CPSs will take responsibility for decision-making in various social fields, including autonomous health monitoring and repair of road infrastructures. The safety assessment of buildings has been supported by professional wisdom that has not necessarily been recorded, so future CPSs must systematize professional knowledge within civil engineering without depending on any language and rule bases. We believe the deep sensing approach may be one of the solutions for this knowledge-encoding problem, but there are many issues to be solved urgently, even within the research field of bridge health monitoring. Future work will include the following three major issues.

Generalized Traffic Surveillance System In this dissertation, TSS implementation described in Chapter 4 completely depended on the installation situations of the surveillance cameras, including angle, focal length, frame rate, and resolution. This dependence may be a severe limitation for the application of the deep sensing approach to other bridges, because the installation points of a camera may be limited by geometric and political reasons, e.g., obstacles and neighborhood privacy protection requirements. Therefore, we must develop a generalized framework of traffic surveillance that can be optimized automatically to individual bridges. One possibility would be for the vehicle properties obtained by TSS to be fed to the deep sensing models completely inside the neural paths. In fact, the SPS presented in Chapter 7 is one such example. Unlike the vehicle detection tasks in Chapter 5 and the anomaly-scoring task in Chapter 6, strain prediction required only the times of vehicle appearance at the bridge entrance. By always handling vehicle properties in a feature space, TSS implementation may be simplified, thus improving generalizability.

Damage Detection based on Acceleration As described in Chapter 5, a weakness of a strain sensor is that it tends to peel off from the bridge material, thereby requiring regular inspection. It would therefore be preferable to develop a damage detection system using accelerometers rather than strain sensors. For now, the application of acceleration data in this dissertation is limited to detecting heavy vehicles because acceleration sensors have poor spatial locality and tend to be influenced by distant vehicles. As a result, the deep sensing models failed to count axles accurately, and it was impossible to model bridge dynamics using acceleration data because axle load information may be undetectable from acceleration data. However, we believe this model may become a definitive approach to providing a durable, inexpensive health-monitoring system.

Fully Neural Vehicle Weighing in Motion Although we demonstrated the potential of a single-sensor BWIM system in Chapter 5, the proposed BWIM system required test runs using a vehicle with known axle weights for calibration. Test runs involving traffic closures on real bridges can frequently be unrealistic, and we must develop a method for collecting influence lines by means of a fully automated mechanism. This may be addressed by embedding influence lines inside the deep sensing models. In other words, a fully neural BWIM system may be a definitive solution to the problem. To realize this, we must develop a framework for collecting the ground truth of axle weight data. Fortunately, in the case of expressways, we can utilize axle load meters installed at tollgates. Generally, the loading and unloading of a vehicle running on an expressway may be ignorable, and axle weighing data sampled at tollgates may be sufficiently accurate for the purpose of training deep sensing models. By tracking individual vehicles using surveillance cameras installed at tollgates and target bridges, fully neural BWIM systems may be realized.

Bibliography

[1] C. Bao, Y. Wu, H. Ling, and H. Ji. Real time robust l1 tracker using accelerated proximal gradient approach. In *CVPR*, 2012.

[2] N. Bicanic and H.-P. Chen. Damage identification in framed structures using natural frequencies. *IJNME*, 40(23), 1997.

[3] Z. Bing, W. Tiesheng, and L. Shanshan. The research of monitoring model for bridge running. In *ICSSEM*, 2011.

[4] M. Braham and M. Van Droogenbroeck. Deep background subtraction with scene-specific convolutional neural networks. In *ICSSIP*, 2016.

[5] J. Bromley, I. Guyon, Y. LeCun, E. Sackinger, and R. Shah. Signature verification using a ”siamese” time delay neural network. In *NIPS*, 1993.

[6] J. Canny. A computational approach to edge detection. *TPAMI*, 8(6), 1986.

[7] D. Cantero and A. Gonzalez. Bridge damage detection using weigh-in-motion technology. *JBE*, 20(5), 2015.

[8] M. S. Cao, G. G. Sha, Y. F. Gao, and W. Ostachowicz. Structural damage identification using damping: a compendium of uses and features. *SMS*, 26(4), 2017.

[9] R. Caruana. Multitask learning. *Machine Learning*, 28(1), 1997.

[10] Z. Chen, Q. Cai, Y. Lei, and S. Zhu. Damage detection of long-span bridges using stress influence lines incorporated control charts. *SCTS*, 57(9), 2014.

[11] P. Cornwell, C. R. Farrar, S. W. Doebling, and H. Sohn. Environmental variability of modal properties. *Experimental Techniques*, 23(6), 1999.

[12] J. Dai, K. He, Y. Li, S. Ren, and J. Sun. Instance-sensitive fully convolutional networks. In *ECCV*, 2016.

[13] W. Dai, C. Dai, S. Qu, J. Li, and S. Das. Very deep convolutional neural networks for raw waveforms. In *ICASSP*, 2017.

[14] C. Donahue, J. McAuley, and M. Puckette. Adversarial audio synthesis. In *ICLR*, 2018.

[15] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, 1995.

[16] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and

an application to boosting. *JCSS*, 55(1), 1997.

[17] K. Funahashi. On the approximate realization of continuous mapping by neural networks. *Neural Networks*, 2(3), 1989.

[18] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2013.

[19] X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *AISTATS*, 2011.

[20] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, 2014.

[21] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[22] K. He, X. Zhang, S. Ren, and J. Sun. Identity mapping in deep residual networks. In *ECCV*, 2016.

[23] W. He, L. Deng, H. Shi, C. S. Cai, and Y. Yu. Novel virtual simply supported beam method for detecting the speed and axles of moving vehicles on bridges. *JBE*, 22(4), 2017.

[24] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *CoRR*, 2014.

[25] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313, 2006.

[26] Y. Hua and T. K. Sarkar. Matrix pencil method for estimating parameters of exponentially damped/undamped sinusoids in noise. *ASSP*, 38(5), 1990.

[27] Y. Huang, C. Zhu, Y. Ye, and Y. Xiao. Damage detection of arch structure by using deflection influence line. In *SEEIE*, 2016.

[28] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017.

[29] C. Jin, J. Li, S. Jangand X. Sun, and R. Christenson. Structural damage detection for in-service highway bridge under operational and environmental variability. In *SPIE*, 2015.

[30] H. Kalhori, M. M. Alamdari, X. Zhu, B. Samali, and S. Mustapha. Non-intrusive schemes for speed and axle identification in bridge-weigh-in-motion systems. *MST*, 28(2), 2017.

[31] T. Kawakatsu, K. Aihara, A. Takasu, and J. Adachi. Adversarial media-fusion approach to strain prediction for bridges. In *ICPRAM*, 2019.

[32] T. Kawakatsu, A. Kakitani, K. Aihara, A. Takasu, and J. Adachi. Traffic surveillance system for bridge vibration analysis. In *IICPS*, 2017.

[33] T. Kawakatsu, A. Kinoshita, K. Aihara, A. Takasu, and J. Adachi. Adversarial spiral

learning approach to strain analysis for bridge damage detection. In *DaWaK*, 2018.

[34] T. Kawakatsu, A. Kinoshita, K. Aihara, A. Takasu, and J. Adachi. Deep sensing approach to single-sensor bridge weighing in motion. In *EWSHM*, 2018.

[35] T. Kawakatsu, A. Kinoshita, K. Aihara, A. Takasu, and J. Adachi. Deep sensing approach to single-sensor vehicle weighing system on bridges. *IEEE Sensors*, 19(1), 2019.

[36] M. G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1-2), 1938.

[37] D. P. Kingma and J. Ba. Adam: a method for stochastic optimization. In *ICLR*, 2015.

[38] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

[39] Y. Kobayashi, C. Miki, and A. Tanabe. Long-term monitoring of traffic loads by automatic real-time weigh-in-motion. *JSCE*, 2004(773):99–111, 2004.

[40] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI*, 1995.

[41] K. Y. Koo, J. M. W. Brownjohn, D. I. List, and R. Cole. Structural health monitoring of the tamar suspension bridge. *SCHM*, 20(4), 2013.

[42] S. K. Leming and H. L. Stalford. Bridge weigh-in-motion system development using superposition of dynamic truck/static bridge interaction. In *ACC*, 2003.

[43] S. Li, H. Li, Y. Liu, C. Lan, W. Zhou, and J. Ou. SMC structural health monitoring benchmark problem using monitored data from an actual cable-stayed bridge. *SCHM*, 21(2), 2014.

[44] X. Li, Z. Han, L. Wang, and H. Lu. Visual tracking via random walks on graph model. *IEEE Transactions on Cybernetics*, 46(9), 2016.

[45] Y. Liu and X. Wang. The application of BP neural network in cable-stayed bridge construction monitoring. In *ICCIS*, 2010.

[46] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[47] M. Lydon, S. Taylor, D. Robinson, A. Mufti, and E. OBrien. Recent developmenets in bridge weigh in motion (bwim). *CSHM*, 6, 2015.

[48] Y. Ma and H. Bi. Cable tension monitoring of suspender arch bridges during cable tension adjustment stage basis on neural network algorithm. In *RSETE*, 2011.

[49] A. L. Maas. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, 2013.

[50] F. Magalhaes, A. Cunha, and E. Caetano. Vibration based structural health monitoring of an arch bridge: From automated OMA to damage detection. *MSSP*, 28, 2012.

[51] A. Malekjafarian, P. J. McGetrick, and E. J. OBrien. A review of indirect bridge monitoring using passing vehicles. *Shock and Vibration*, 2015.

[52] MariaDB Foundation. MariaDB. `https://mariadb.org`.

[53] X. Mei and H. Ling. Robust visual tracking using l1 mimimization. In *ICCV*, 2009.

[54] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013.

[55] B. Mohamed and E. Tahar. Remote sensing of damage bridge structure of free vibration by using finite element method based on subspace fitting. In *ICIS*, 2017.

[56] B. Mohamed, E. Tahar, and R. Houria. Remote sensing of damage bridge structure of free vibration by using finite element method based on jerk-energy. In *SERA*, 2017.

[57] MongoDB, Inc. MongoDB. `https://www.mongodb.com`.

[58] A. C. Neves, I. Gonzalez, J. Leander, and R. Karoumi. Structural health monitoring of bridges: a model-free ANN-based approach to damage detection. *CSHM*, 7(5), 2017.

[59] A. C. Neves, I. Gonzalez, J. Leander, and R. Karoumi. A new approach to damage detection in bridges using machine learning. In *EVACES*, 2018.

[60] NVIDIA Corporation. CUDA. `http://developer.nvidia.com/cuda`.

[61] E. J. OBrien and A. Malekjafarian. Use of a passing vehicle to scan the fundamental bridge frequencies: An experimental verification. *Engineering Structures*, 27(13), 2005.

[62] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier gans. In *ICML*, 2017.

[63] T. Ojio, C. H. Carey, E. J. OBrien, C. Doherty, and S. E. Taylor. Contactless bridge weigh-in-motion. *JBE*, 21(7), 2016.

[64] R. P. Palanisamy and S.-H. Sim. Bridge scour monitoring using extended kalman filter. In *AESE*, 2015.

[65] C.-D. Pan, L. Yu, and H.-L. Liu. Identification of moving vehicle forces on bridge structures via moving average tikhonov regularization. *SMS*, 26, 2017.

[66] PASCAL. VOC. `http://host.robots.ox.ac.uk/pascal/VOC/`.

[67] P. H. O. Pinheiro, R. Collbert, and P. Dollár. Learning to segment object candidates. In *NIPS*, 2015.

[68] Preferred Networks, Inc. Chainer. `http://chainer.org`.

[69] Project Jupyter. Jupyter. `https://jupyter.org`.

[70] S. Quansheng, Y. Haiying, G. Xiaoguang, W. Jiawei, and W. Tong. Application of kalman's filtering method in construction control for cable replacement of the cable-stayed bridge. In *ICECE*, 2010.

[71] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016.

[72] S. Reddi, S. Kale, and S. Kumar. On the convergence of adam and beyond. In *ICLR*, 2018.

[73] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: Towards real-time object

detection with region proposal networks. In *NIPS*, 2015.

[74] J. Richardson, S. Jones, A. Brown, E. Obrien, and D. Hajializadeh. On the use of bridge weigh-in-motion for overweight truck enforcement. *JVS*, 21(2), 2014.

[75] A. R. Rivera, M. Murshed, J. Kim, and O. Chae. Background modeling through statistical edge-segment distributions. *TCSVT*, 23(8), 2013.

[76] Road Bureau, Ministry of Land, Infrastructure, Transport and Tourism, Government of Japan. Road maintenance in japan: Problems and solutions. `http://www.mlit.go.jp/road/road_e/pdf/RoadMaintenance.pdf`, April 2015.

[77] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *IJCV*, 77(1), 2008.

[78] D. E. Rumelhard, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323, 1986.

[79] T. K. Sarkar and O. Pereira. Using the matrix pencil method to estimate the parameters of a sum of complex exponentials. *IEEE Antennas and Propagation Magazine*, 37(1), 1995.

[80] Japan Science and Technology Agency. Cross-ministerial strategic innovation promotion (SIP) program. `http://www.jst.go.jp/sip/k07.html`.

[81] H. Sekiya, K. Kubota, and C. Miki. Simplified portable bridge weigh-in-motion system using accelerometers. *JBE*, 23(1), 2018.

[82] A. A. Shah, B. S. Chowdhry, J. Daudpoto, and I. Ali. Transient structural health monitoring of the test bridges using finite element method. In *IMTIC*, 2018.

[83] H. H. Shahri, G. Namata, S. Navlakha, A. Deshpande, and N. Roussopoulos. A graph-based approach to vehicle tracking in traffic camera video streams. In *DMSN*, 2007.

[84] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[85] N. Sirikuntamat, S. Satoh, and T. H. Chalidabhongse. Vehicle tracking in low hue contrast based on CAMShift and background subtraction. In *JCSSE*, 2015.

[86] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 15:1929–1958, 2014.

[87] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. In *CVPR*, volume 2, 1999.

[88] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *NIPS*, 2014.

[89] K. Tateishi, H. Takenouchi, and C. Miki. Mechanism for developing local stress at the connection details in steel bridge structures. *JSCE*, 1995(507):109–119, 1995.

[90] E. Toropov, L. Gui, S. Zhang, S. Kottur, and J. M. F. Moura. Traffic flow from a low frame rate city camera. In *ICIP*, 2015.

[91] J. R. R. Uijlings, K. W. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *IJCV*, 104(2), 2013.

[92] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: a neural image caption generator. In *CVPR*, 2015.

[93] P. Viola and M. J. Jones. Robust real-time face detection. *IJCV*, 57(2), 2004.

[94] L. Wang, W. Ouyang, X. Wang, and H. Lu. Visual tracking with fully convolutional networks. 2015.

[95] B. Wu, H. Lu, B. Chen, and Z. Gao. Study on finite element model updating in highway bridge static loading test using spatially-distributed optical fiber sensors. *Sensors*, 17(7), 2017.

[96] C. Xiao and Z. Fang. Research on multi-sensor information fusion algorithm with sensor fault diagnosis. In *ICIICII*, 2016.

[97] S. Yang, L. Xie, X. Chen, X. Lou, X. Zhu, D. Huang, and H. Li. Statistical parametric speech synthesis using generative adversarial networks under a multi-task learning framework. In *ASRU*, 2017.

[98] Y. Yao, L. Rosasco, and A. Caponnetto. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2), 2007.

[99] Y. Yu, CS Cai, and L. Deng. State-of-the-art review on bridge weigh-in-motion technology. *Advances in Structural Engineering*, 19(9), 2016.

[100] Y. Yu, CS. Cai, and L. Deng. Vehicle axle identification using wavelet analysis of bridge global responses. *JVC*, 23(17), 2017.

[101] R. Zaurin and F. N. Catbas. Integration of computer imaging and sensor data for structural health monitoring of bridges. *SMS*, 19(1), 2010.

[102] R. Zaurin and F. N. Catbas. Structural health monitoring using video stream, influence lines, and statistical analysis. *SHM*, 10(3), 2011.

[103] B. Zhang, H. Jiang, and L. Dong. Classification of EEG signal by WT-CNN model in emotion recognition system. In *ICCICC*, 2017.

[104] B. Zhang, C. Quan, and F. Ren. Study on CNN in the recognition of emotion in audio and images. In *ICIS*, 2016.

[105] L. Zhang, Z. Sun, C. Zhang, F. Dong, and P. Wei. Numerical investigation of the dynamic responses of long-span bridges with consideration of the random traffic flow based on the intelligent ACC-BPNN model. *IEEE Access*, 6:28520–28529, 2018.

[106] L. Zhang, G. Zhou, Y. Han, H. Lin, and Y. Wu. Application of internet of things technology and convolutional neural network model in bridge crack detection. *IEEE*

Access, 6:39442–39451, 2018.

[107] H. Zhao, N. Uddin, E. J. OBrien, and X. Shao. Identification of vehicular axle weights with a BWIM system considering transverse distribution of wheel loads. *JBE*, 19(3), 2014.

Publications

Original Articles

[1] T. Kawakatsu, A. Kinoshita, K. Aihara, A. Takasu, and J. Adachi. Deep sensing approach to single-sensor vehicle weighing system on bridges. *IEEE Sensors*, 19(1):243–256, 2019.

[2] K. Aihara, A. Takasu, T. Kawakatsu, A. Kinoshita, H. Imura, and J. Adach. Integrated data management platform for efficient long-term monitoring of structural health. Rejected by JSCE Structural Engineering, 2019.

International Conferences

[1] T. Kawakatsu, A. Kakitani, K. Aihara, A. Takasu, and J. Adachi. Traffic surveillance system for bridge vibration analysis. In *International Workshop on Information Integration in Cyber Physical Systems (IICPS)*, Aug. 2017.

[2] T. Kawakatsu, A. Kinoshita, K. Aihara, A. Takasu, and J. Adachi. Deep sensing approach to single-sensor bridge weighing in motion. In *European Workshop on Structural Health Monitoring (EWSHM)*, Jul. 2018.

[3] T. Kawakatsu, A. Kinoshita, K. Aihara, A. Takasu, and J. Adachi. Deep sensing approach to vehicle detection from bridge vibration. In *International Workshop on Information Search, Integration, and Personalization (ISIP)*, May 2018. (Oral presentation only).

[4] T. Kawakatsu, A. Kinoshita, K. Aihara, A. Takasu, and J. Adachi. Adversarial spiral learning approach to strain analysis for bridge damage detection. In *International Conference on Big Data Analytics and Knowledge Discovery (DaWaK)*, Sep. 2018.

[5] T. Kawakatsu, K. Aihara, A. Takasu, and J. Adachi. Adversarial media-fusion approach to strain prediction for bridges. In *International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, Feb. 2019.

Miscellaneous Publications

[1] T. Kawakatsu, A. Kinoshita, A. Takasu, and J. Adachi. Divide-and-conquer parallelism for learning mixture models. *Transactions on Large-Scale Data and Knowledge-Centered Systems (TLDKS)*, 28, 2016.

[2] T. Kawakatsu, A. Kinoshita, A. Takasu, and J. Adachi. Highly efficient parallel framework: A divide-and-conquer approach. In *International Conference on Database and Expert Systems Applications (DEXA)*, 2015.