

Using Minimal GRU Networks for Cross-Domain Damage Detection  
in Structural Health Monitoring

Riley Starling

Thesis Advisor  
Prof. Homayoon Beigi

Submitted in partial fulfillment of the  
requirements for the degree of  
Master in Mechanical Engineering  
under the Executive Committee  
of the Fu Foundation School of Engineering and Applied Science

COLUMBIA UNIVERSITY

## **Abstract**

Using Minimal GRU Networks for Cross-Domain Damage Detection  
in Structural Health Monitoring

Riley Starling

Identifying damage patterns in in-service structures for early maintenance and prevention remains a challenging task due to the limited availability of reliable, labeled operational data.

Experimental datasets help bridge this gap by providing sufficient data to train reliable models; however, for practical deployment, these models must generalize effectively to out-of-domain structures and conditions. In this work, we address the generalization challenge by training neural network models on experimental structural data and evaluating their performance on the Z24 Bridge Benchmark. Structural vibrations are represented using cepstral coefficients, which capture underlying frequency characteristics and are used to extract damage-sensitive features.

This study investigates two models: the first, a Minimal GRU (MinGRU), leverages a simplified recurrent architecture to efficiently learn temporal dependencies, while the second, a Bidirectional Minimal GRU (BiMinGRU), extends this approach to incorporate forward and backward processing to capture temporal dependencies in both directions. Model embeddings are analyzed using probabilistic linear discriminant analysis and evaluated through hypothesis-based damage detection. The models are initially evaluated on damage classification within the training domain, followed by cross-domain testing, in which deviations from the baseline condition are used to determine the damage state.

## Table of Contents

Acknowledgments . . . . .	v
Nomenclature . . . . .	vi
Chapter 1: Introduction . . . . .	1
1.1 Background . . . . .	2
1.1.1 Recurrent Neural Networks . . . . .	2
1.1.2 Cepstral Coefficients . . . . .	5
Chapter 2: Methods . . . . .	8
2.1 Datasets . . . . .	8
2.1.1 Laboratory Experiments . . . . .	8
2.1.2 In-Service Structures . . . . .	9
2.2 Model Architecture . . . . .	9
2.2.1 MinGRU . . . . .	9
2.2.2 BiMinGRU . . . . .	11
2.3 Training and Classification . . . . .	12
2.3.1 Multi-Task Heads . . . . .	14
2.3.2 Feature Space Analysis . . . . .	15
2.3.3 Likelihood-Based Scoring . . . . .	16

2.4	Zero-Shot Evaluation . . . . .	17
Chapter 3:	Results . . . . .	20
3.0.1	Supervised In-domain . . . . .	20
3.0.2	Unsupervised Cross-domain . . . . .	23
3.0.3	Supervised Cross-domain . . . . .	27
Chapter 4:	Discussion . . . . .	30
4.1	Embedding Separability: Metrics and Visualization . . . . .	30
4.2	Generalization Under Domain Shift . . . . .	31
4.3	Comparison of MinGRU and BiMinGRU . . . . .	32
Chapter 5:	Conclusion . . . . .	34
References	. . . . .	36

## List of Figures

1.1	GRU Architecture Diagram . . . . .	3
1.2	MinGRU Architecture . . . . .	5
2.1	Z24 Bridge Structural Configuration . . . . .	10
2.2	Class Distribution . . . . .	14
3.1	PCA of PLDA Training Embeddings (Supervised In-Domain) . . . . .	21
3.2	PCA of PLDA Task A Training Embeddings (Supervised In-Domain) . . . . .	22
3.3	Confusion Matrix (Supervised In-Domain) . . . . .	23
3.4	PCA projection of PLDA test embeddings for supervised in-domain setting . . . . .	24
3.5	PCA of PLDA Z24 Embeddings (Unsupervised Cross-Domain) . . . . .	24
3.6	PCA of PLDA Z24 Extreme Embeddings (Unsupervised Cross-Domain) . . . . .	25
3.7	LLR Scores (Unsupervised Cross-Domain) . . . . .	26
3.8	Confusion Matrix (Unsupervised Cross-Domain) . . . . .	27
3.9	PCA of PLDA Z24 Embeddings (Supervised Cross-Domain) . . . . .	28
3.10	PCA of PLDA Z24 Extreme Embeddings (Supervised Cross-Domain) . . . . .	28
3.11	Confusion Matrix (Supervised Cross-Domain) . . . . .	29

## List of Tables

2.1	Z24 Bridge Damage Cases . . . . .	11
3.1	Summary of Model Capacity . . . . .	20
3.2	Performance Metrics (Supervised In-Domain) . . . . .	23
3.3	Performance Metrics (Unsupervised Cross-Domain) . . . . .	26
3.4	Performance Metrics (Supervised Cross-Domain) . . . . .	29

## **Acknowledgements**

I want to express my gratitude to my advisor, Professor Beigi, for his guidance, mentorship, and invaluable feedback throughout this research. His instruction and insight were instrumental in shaping both the technical direction and overall quality of this work.

I am also grateful to Professor Raimondo Betti for serving as a co-advisor and for his thoughtful feedback, expertise, and support, which greatly strengthened this study.

I want to thank Dr. Azin Mehrjoo for her assistance with coding and project organization, and Mr. Kyle Hom for his help in managing and preparing the extracted datasets. Their contributions were essential to the successful completion of this work.

Finally, I would like to thank all the members of the Structural Health Monitoring group for their support, collaboration, and encouragement throughout this semester.

## Nomenclature

$\mathbf{I}$	Identity matrix
$\mathbf{u}$	Class in latent-space class
$\mathbf{v}$	Class in original data space
$\mathcal{N}$	Normal Distribution
$\Psi$	Covariance matrix
$\tau$	EER threshold
$\tau$	Validation-derived threshold
LLR	Log likelihood ratio
$A$	Transformer matrix (original to latent space)
$a_i$	Average Intra-Cluster Distance
$b_i$	Average Nearest-Cluster Distance
$H$	Hypothesis
$K$	Number of Classification Heads
$N_e$	Number of Embeddings
$P$	Probability
$R$	Likelihood ratio
$S$	Silhouette Score
$w$	Window
$W_k, b_k$	Weights and biases
$x_z$	Z24 embedding
$\check{H}_m$	Filtered frequency spectrum

$\mathbf{c}_d$	Cepstral coefficient vector
$\mathbf{m}$	Mean
$\mathbf{W}, \mathbf{U}$	Learned weights
$\mathcal{M}_{mk}$	Mel filterbank
$\sigma$	Logistic sigmoid function
$\tilde{h}$	Candidate hidden state
$\varphi$	Hyperbolic tangent function
$c_d$	Cepstral coefficient components
$C_m$	Log-energy spectrum
$h$	Hidden state
$H_k$	Frequency spectrum produced by FFT
$r$	Reset gate
$s_n$	$n$ -th sample of a signal frame
$t$	time
$w_H$	Hanning window function
$x$	Input state
$y_{\text{logits}}$	Logits
$z$	Update gate

## Chapter 1: Introduction

Early damage detection is a key challenge in Structural Health Monitoring (SHM) and relies on reliable long-term data recorded on structures via sensors. Early intervention is crucial for servicing and maintaining infrastructures before disasters occur. There are numerous methods for detecting damage and evaluating its severity, including monitoring dynamic properties and local visual inspection.

A structure is considered damaged when its structural state deviates from its baseline, or healthy, state [1]. In SHM, damage is commonly identified by its existence, location, severity, and impact on structural life span [2]. Addressing these identifiers is often formulated using this four-step statistical pattern recognition framework: (1) operational assessment, (2) data acquisition and processing, (3) feature selection and extraction, and (4) model development for decision making [3]. Numerous attempts have been made to detect damage by monitoring deviations in structural vibrations, many of which use signal processing techniques, including spectral [4, 5] and wavelet-based [6, 7] analyses. Cepstral analysis [8, 9] is an additional method successfully applied to vibration-based damage detection due to its sensitivity to structural changes.

These signal processing techniques commonly lead to classification by integrating deep learning [10, 11]. However, these methods rely on access to substantial, comprehensive data from both damaged and undamaged structures. Data from in-service structures, damaged structures, simulations, and experimental setups are all methods of interest, with varying availability. Intentionally damaging an in-service structure has significant safety concerns, and structures during unplanned major damage events are rarely monitored. Environmental and operational variability can also introduce changes in measured responses that may be mistaken for damage. Thus, pretraining deep learning networks on experimental datasets via transfer learning has emerged as a new approach to mitigate unreliable data [12].

In this study, damage from the Z24 Bridge progressive damage tests [13] are analyzed using cepstral analysis and a minimal gated recurrent unit (MinGRU) [14]. The model will be pretrained on two experimental datasets, focusing on learning patterns of damage type and severity.

## 1.1 Background

### 1.1.1 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) once dominated the field for processing sequential data before the rise of transformers. The output of RNNs is determined by the input and the hidden state, where the hidden state stores information from previous inputs. In essence, previous steps inform new predictions. A common challenge of RNNs is vanishing gradients, which limit the ability to learn long-term dependencies. Long Short-Term Memory (LSTM) networks addressed this issue by introducing new components, the memory cell and the gates, to retain information more efficiently [15]. The memory cell ensures the controller stores long-term information, while the three gates control the flow of information. The input, the output, and the forget gates continuously determine when information is added, sent, or forgotten by the memory cell [16]. As such, there are two layers of memory, the cell state, which acts as the controller's long-term memory, and the hidden state, as the short-term memory and output [14].

Introduced in 2014, Gated Recurrent Units (GRUs) offered an additional, streamlined way of addressing the vanishing gradient problem. GRUs employ two gates, the reset and update, that control how much information is dropped or retained from one hidden state to the next [17]. Unlike the LSTM, there is no separate state for long-term memory, and instead, the hidden state is a unified memory. This unification enables GRUs to achieve faster computation times while maintaining competitive results.

GRUs are governed by the following equations:

$$\text{Hidden State: } h_j^{(t)} = z_j h_j^{(t-1)} + (1 - z_j) \tilde{h}_j^{(t)} \quad (1.1)$$

$$\text{Candidate Hidden State: } \tilde{h}_j^{(t)} = \varphi([\mathbf{W}\mathbf{x}]_j + [\mathbf{U}(\mathbf{r} \odot \mathbf{h}_{t-1})]_j) \quad (1.2)$$

$$\text{Update Gate: } z_j = \sigma([\mathbf{W}_z\mathbf{x}]_j + [\mathbf{U}_z\mathbf{h}_{t-1}]_j) \quad (1.3)$$

$$\text{Reset Gate: } r_j = \sigma([\mathbf{W}_r\mathbf{x}]_j + [\mathbf{U}_r\mathbf{h}_{t-1}]_j) \quad (1.4)$$

where  $h_j^{(t)}$  and  $\tilde{h}_j^{(t)}$  are the  $j$ -th hidden unit of the hidden state and the candidate hidden state, respectively. More,  $\mathbf{x}$  is the input,  $\mathbf{h}_{t-1}$  is the previous hidden state,  $\mathbf{W}$  and  $\mathbf{U}$  are learned weight matrices, while  $\sigma$  is the logistic sigmoid function and  $\varphi$  is a hyperbolic tangent function. Hence, the update gate computes how much information from the previous hidden state  $\mathbf{h}_{t-1}$  is remembered and how much information from the candidate hidden state  $\tilde{h}_j^{(t)}$  should be added to the hidden state. Additionally, the reset gate determines how much of the previous hidden state is used in the computation of the candidate hidden state. An illustration of this process is summarized in Figure 1.1.

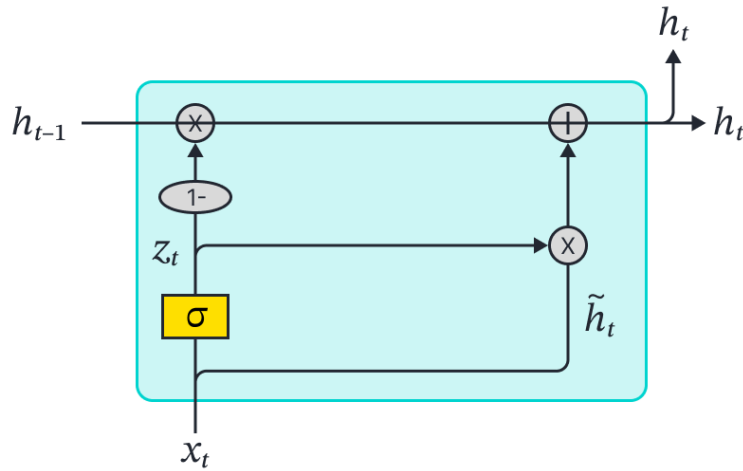


Figure 1.1: GRU architecture illustration from [18].

Like LSTMs, the GRU architecture is inherently sequential due to the use of backpropagation through time (BPTT); this means processing long sequences is slow, and there remains a potential for vanishing gradients, albeit reduced relative to the vanilla RNNs. However, in 2017, Trans-

formers were introduced and quickly replaced RNNs as the field’s frontrunner. Transformers are a feed-forward neural network model that allow for parallelization due to its self-attention mechanism, which means parts of sequences can be processed simultaneously instead of sequentially [19].

## MinGRU

Recently, however, Feng et al. (2024) [14] introduced two novel simplified versions of LSTM and GRU that can train in parallel. These versions, called minLSTM and minGRU, are obtained by removing previous hidden state dependencies and constraints on the output range [14]. Currently, parallel scan cannot be applied to standard GRUs due to its sequential nature, therefore if one removes the reliance on the previous hidden state  $\mathbf{h}_{t-1}$  from Equations 1.2 and 1.3, BPTT is no longer required and parallel training is possible [14]. Further, by removing  $\mathbf{h}_{t-1}$  from the candidate hidden state, the need for a reset gate (Equation 1.4) is also eliminated. More, in GRU the hyperbolic tangent function  $\varphi$  is necessary to reduce vanishing gradients caused by the previous hidden state in the update gate. Since this dependency was just eliminated, there is no need for  $\varphi$ , and the governing equations can be further reduced. The result of these modifications is illustrated in Figure 1.2 and governed as follows:

$$\text{Hidden State: } h_j^{(t)} = z_j h_j^{(t-1)} + (1 - z_j) \tilde{h}_j^{(t)} \quad (1.5)$$

$$\text{Candidate Hidden State : } \tilde{h}_j^{(t)} = \varphi([\mathbf{W}\mathbf{x}]_j) \quad (1.6)$$

$$\text{Update Gate : } z_j = [\mathbf{W}_z \mathbf{x}]_j \quad (1.7)$$

This minimal model is significantly more efficient than its parent GRU because it can be trained in parallel, has a lower model capacity, and trains faster. MinGRU requires  $O(2d_h d_x)$  parameters while standard GRUs require  $O(3d_h(d_x + d_h))$  parameters, where  $d_x$  is the size of the input state and  $d_h$  is the size of the hidden. It has also performed comparably against, and in many cases

outperformed, other algorithms in Reinforcement Learning and Selective Copy Task experiments [14]; however, its application to SHM has received limited attention and is investigated in this study.

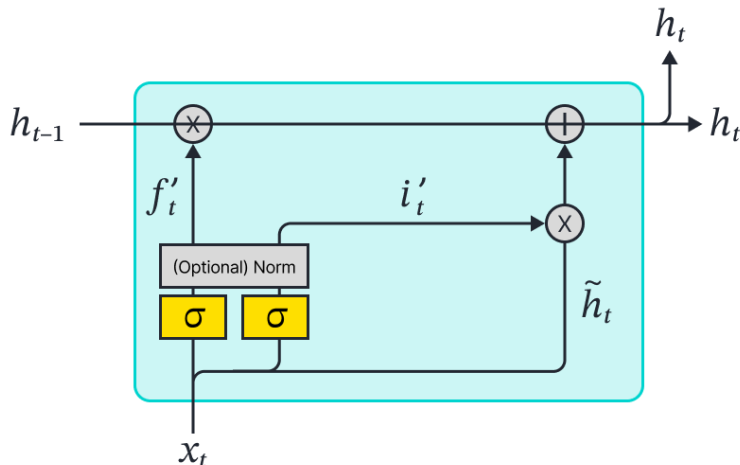


Figure 1.2: MinGRU architecture illustration from [18].

### 1.1.2 Cepstral Coefficients

The cepstral coefficients of the raw accelerometer signals are our chosen features for evaluating damage patterns. Cepstral coefficients are a representation of a frequency spectrum; a subset of coefficients is chosen to represent the most important information from a signal [20]. Typically, cepstral coefficients represent the spectra of a sound signal [21]. Still, they have also been shown to be well-suited for SHM application [12] and can be used to extract damage-sensitive features from vibrational data [9].

Leveraging the raw acceleration signals collected across multiple datasets, we segment the continuous time-series data into overlapping frames to achieve quasi-stationary conditions and prevent discontinuities. Each frame represents a short, fixed-length segment of the signal, and the overlap ensures smoother transitions and better capture of dynamic changes. From each frame, we are able to extract cepstral coefficient features to serve as inputs for model training. This process is outlined below, for greater detail reference Chapter 5 in [8].

## Feature Extraction

We start by segmenting  $L$  frames of the signal into  $N$  samples, representing the  $n^{th}$  sample of the  $l^{th}$  frame as  $s_n^l$ . Then, after multiplying each frame by the Hanning window function  $w_H(n)$  to minimize spectral leakage, we perform a Discrete Fourier Transform (DFT) [8] on the windowed sample using Fast Fourier Transform (FFT) [8]. The result of the FFT with window  $w(n)$  is as follows [8]:

$$w_H(n) = 0.5 \left( 1 - \cos \left( \frac{2\pi n}{N-1} \right) \right) \quad (1.8)$$

$$H_k = \sum_{n=0}^{N-1} s_n^l w(n) e^{-i\frac{2\pi kn}{N}} \quad (1.9)$$

where,  $n \in \{0, 1, \dots, N-1\}$  and the frequency domain index  $k = \{0, 1, \dots, N-1\}$ . Next, taking the magnitude of this result provides an estimate of the sequence's frequency content. Further, the filterbank  $\mathcal{M}_{mk}$  processes the high-resolution spectrum by splitting it into multiple frequency bands, enabling noise suppression and complexity reduction. Here,  $m = \{0, 1, \dots, M-1\}$  is the critical filter index and  $M$  is the total number of filters. Normally,  $\mathcal{M}_{mk}$  is arranged using Mel-frequency bins [8], but for structural applications, changes are more effectively captured with linearly spaced frequencies. Thus,  $\mathcal{M}_{mk}$  is spaced along linear-frequency bins.

$$|H_k| = \sqrt{\text{Re}(H_k)^2 + \text{Im}(H_k)^2} \quad (1.10)$$

$$\check{H}_m = \mathcal{M}_{mk} |H_k| \quad (1.11)$$

Taking the logarithm of the filtered spectrum compresses the dynamic range, thereby improving the visualization of features across a range of amplitudes. This produces the log-energy spectrum  $C_m$ , which indicates the amount of energy in each filterband on a logarithmic scale [8].

$$C_m = \log(|\check{H}_m|^2) \quad (1.12)$$

To obtain the cepstral coefficients, one option is to compute the inverse Fourier transform of the log-energy output  $C_m$ , however, the result is complex. Instead, we apply the inverse Discrete Cosine Transform (DCT) [8] to obtain real-valued coefficients. Thus, the LFCCs are defined as follows using DCT:

$$c_d = \sum_{m=0}^{M-1} a_m C_m \cos\left(\frac{\pi(2d+1)m}{2M}\right) \quad (1.13)$$

where,

$$a_m = \begin{cases} \frac{1}{M} & \text{for } m = 0 \\ \frac{2}{M} & \text{for all } m > 0 \end{cases} \quad (1.14)$$

The cepstral coefficient components  $c_d$  comprise the vector  $\mathbf{c}_d$ , which expresses the signal's spectrum in a compact form that is sensitive to changes in the system's dynamic response [8]. Consequently, these vectors serve as input parameters for the MinGRU.

## Chapter 2: Methods

### 2.1 Datasets

Provided in this section is an overview of the datasets used in this paper. The laboratory experiments are used to train and validate our model. The in-service progressive damage test is used to evaluate model generalization.

#### 2.1.1 Laboratory Experiments

##### **LANL Bookshelf**

The LANL bookshelf [22] is a three-story frame constructed of aluminum plates, struts, brackets, and steel bolts. Damage is instigated at various locations by altering bolts and brackets and exciting the structure with banded white noise. Bolt connections are removed, loosened, or tightened to hand-tight, 5 ft-lbs, and 10 ft-lbs. The excitation is applied at three levels of voltage to the base using a shaker inclined at approximately 45 degrees in order to excite the bending and torsion modes. The structure contains 24 accelerometers at the corners of each plate, sampling every 5.12 seconds at a sampling rate of 1600 Hz.

##### **NEESR Reinforced Columns**

The NEESR column study [23] involved a series of experiments on scaled concrete bridge columns to assess the performance of ‘smart aggregate’ sensors, or piezoceramic transducers. These smart aggregates generate electric charge under stress and induce stress when electrically excited, allowing them to both sense and excite concrete columns for damage profiling. In the study, these sensors were affixed to the columns at three heights, and the columns were subjected to hysteretic load cycles and seismic excitations to induce damage. The hysteretic cycles occurred

first, where the column was excited by white noise on a shake table after each cycle, with 9 smart aggregates recording the output. The seismic excitations were performed last at increasing magnitudes, followed by additional white-noise excitation, with 12 smart aggregates recording the output. All recordings are taken at a sampling rate of 20kHz over 5 seconds.

### 2.1.2 In-Service Structures

#### **Z24 Bridge Benchmark**

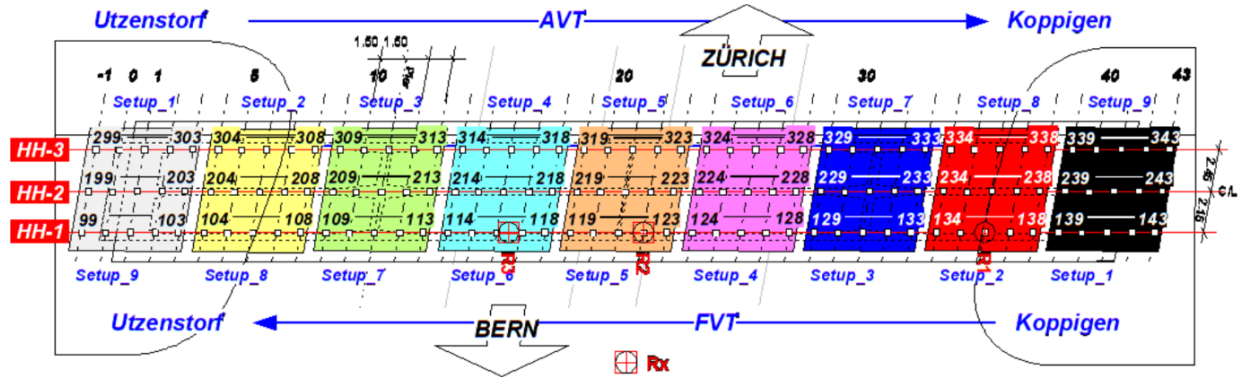
The Z24 bridge was a prestressed box-girder bridge in Switzerland that underwent environmental monitoring and progressive damage tests before its scheduled demolition in 1998 [13]. The Z24 Bridge benchmark contains the Environmental Monitoring System (EMS) [24] and Progressive Damage Test (PDT) [22] datasets. The EMS records long-term ambient vibration under typical operating conditions, while the PDT records ambient and forced vibrations during induced damage cases.

The PDT documents 17 progressive damage cases ranging from minor interventions to severe deterioration, summarized in Table 2.1. For each case, data is recorded sequentially from a grid of sensors distributed on the bridge spans and piers shown in Figure 2.1. Due to limitations in the number of recording channels, data was recorded in nine different setups in different sensor groups. Each sensor was sampled at 100 Hz for 81.92 seconds. For this investigation, only 15 accelerometers from the PDT dataset are evaluated to create a more realistic monitoring scenario with a lower sensor density.

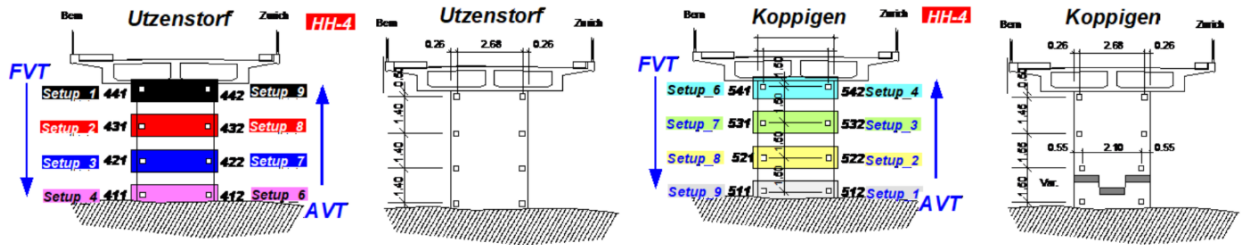
## **2.2 Model Architecture**

### 2.2.1 MinGRU

The basic model is a MinGRU network with a windowed sliding-sequence encoder. The model supports multiple independent classification heads built on the final latent embedding. Each sequence is sliced into overlapping windows  $w \in \{1, \dots, W\}$ , which are processed sequentially. At



(a) Bridge Plan View



(b) Pier Elevations

Figure 2.1: Z24 Bridge Structural Configuration [22]

each timestep within a window, the model applies a single linear layer, producing the candidate hidden state and gate logits. The model uses a log-space recurrence, where the update is computed for the entire window at once using log-space cumulative sums, then exponentiated back to the original space.

$$\log h_t = a_t + \log \left( \sum_{\tau=0}^t e^{\log v_{\tau} - a_{\tau}} \right) \quad (2.1)$$

$$a_t = \sum_{\tau=0}^t \log(1 - z_{\tau}) \quad (2.2)$$

The final hidden state from the previous window,  $h_T^{(w-1)}$ , serves as the initial hidden state  $h_0$  for the next window. The model stores the final timestep for each processed window to form a compressed sequence. The final embedding, or the final hidden state of the final window, is used for classification. The model is capable of multitask learning for  $K$  classification heads. Each

Damage Case	Assigned Location	Damage scenarios
0	-	Undamaged Condition
1	HH4	Installation of pier settlement system Condition
2	HH4	Lowering of pier, 20 mm Condition
3	HH4	Lowering of pier, 40 mm Condition
4	HH4	Lowering of pier, 80 mm Condition
5	HH4	Lowering of pier, 95 mm Condition
6	-	Lifting of pier, tilt of foundation
7	-	New reference condition
8	HH1,2,3	Spalling of concrete at soffit, 12 m2
9	HH1,2,3	Spalling of concrete at soffit, 24 m2
10	Setup1,9	Landslide of 1 m at abutment
11	Setup1,9	Failure of concrete hinge
12	Setup1,9	Failure of 2 anchor heads
13	Setup1,9	Failure of 4 anchor heads
14	HH1,3	Rupture of 2 out of 16 tendons
15	HH1,3	Rupture of 4 out of 16 tendons
16	HH1,3	Rupture of 6 out of 16 tendons

Table 2.1: Damage cases in the Z24 Bridge Benchmark[22]

head is a linear classifier and maps the final embedding to a separate set of class logits. After each forward pass, three outputs are returned: the compressed sequence, the final shared embedding, and a dictionary of classification logits, with one key per head. The algorithm is detailed in Algorithm 1.

### 2.2.2 BiMinGRU

The BiMinGRU architecture extends the MinGRU by processing the input sequence in two directions: a forward MinGRU and a backward MinGRU. Each direction operates independently but uses the same windowed sliding-sequence encoder and the same log-space recurrence formulation as the base MinGRU. For an input sequence sliced into windows, the forward MinGRU processes windows in increasing temporal order, while the backward MinGRU processes windows in reverse order. Each direction produces its own window-final hidden states. Both directions use the same log-space recurrence as in Equation 2.2.

The hidden states from each direction are then concatenated to form a bidirectional embedding.

---

**Algorithm 1** MinGRU Architecture

---

**Input:** Sequence of frames  $\{c_l\}_{l=1}^L$ , windowed into  $W$  overlapping segments.

```
1: for  $w \in \{1, \dots, W\}$  do
2:   Initialize hidden state
3:   if  $w = 1$  then
4:      $h_0 \leftarrow 0$  ▷ first window: set to 0
5:   else
6:      $h_0 \leftarrow h_T^{(w-1)}$  ▷ otherwise: set to  $h_T$  of previous window
7:   end if
8:   for  $t \in \{1, \dots, T\}$  do
9:     Update hidden state:  $h_t \leftarrow \text{MinGRUCell}(c_t, h_{t-1})$ 
10:  end for
11:  Collect final hidden state  $h_T$ 
12:  Classification:  $y_{\text{logits}} \leftarrow W_k h_T + b_k$  for each head  $k$ 
13: end for
```

**Output:** Embeddings  $h_T$ , Probabilities  $\text{softmax}(y_{\text{logits}})$

---

The final bidirectional embedding from the last processed window is provided to all downstream classification heads. As with the MinGRU, the model supports  $K$  classification heads, each mapping the shared bidirectional embedding to task-specific class logits. The algorithm is detailed in Algorithm 2.

### 2.3 Training and Classification

The models are trained on experimental data from several structures, and we perform supervised classification using the training labels. Class imbalances are mitigated by shuffling sequences, having a weighted loss function, and oversampling minority classes with replacement with PyTorch’s `WeightedRandomSampler`. Thus, each batch prepared for the MinGRU has a balanced class distribution, with a representative of each label. For example, without the sampler, the first batch, which contains data from the LANL Bookshelf, had only samples with undamaged data (class 5) for Task B, as seen in Figure 2.2a. However, with the sampler, the same batch now contains a distribution of classes more representative of the dataset, shown in Figure 2.2b

For model training, we aim to minimize the multiclass cross-entropy loss described by equation

---

**Algorithm 2** BiMinGRU Architecture

---

**Input:** Sequence of frames  $\{c_l\}_{l=1}^L$ , windowed into  $W$  overlapping segments.

```
1: for  $w \in \{1, \dots, W\}$  do
2:   Forward: Initialize hidden state
3:   if  $w = 1$  then
4:      $h_0 \leftarrow 0$  ▷ first window: set to 0
5:   else
6:      $h_0 \leftarrow h_T^{(w-1)}$  ▷ otherwise: set to  $h_T$  of previous forward window
7:   end if
8:   Backward: Initialize hidden state
9:   if  $w = 1$  then
10:     $h_0 \leftarrow 0$  ▷ first window: set to 0
11:  else
12:     $h_0 \leftarrow h_t^{(w-1)}$  ▷ otherwise: set to  $h_t$  of previous backward window
13:  end if

14:  Forward pass
15:  for  $t \in \{1, \dots, T\}$  do
16:    Update hidden state:  $h_t \leftarrow \text{MinGRUCell}(c_t, h_{t-1})$ 
17:  end for
18:  Backward pass
19:  for  $t \in \{T, \dots, 1\}$  do
20:    Update hidden state:  $h_{T-t+1} \leftarrow \text{MinGRUCell}(c_t, h_{T-t})$ 
21:  end for

22:  Collect and concatenate final hidden states  $h_T^{bi} \leftarrow [h_T^f || h_1^b]$ 
23:  Classification:  $y_{\text{logits}} \leftarrow W_k h_T^{bi} + b_k$  for each head  $k$ 
24: end for
```

**Output:** Embeddings  $h_T^{bi}$ , Probabilities  $\text{softmax}(y_{\text{logits}})$

---

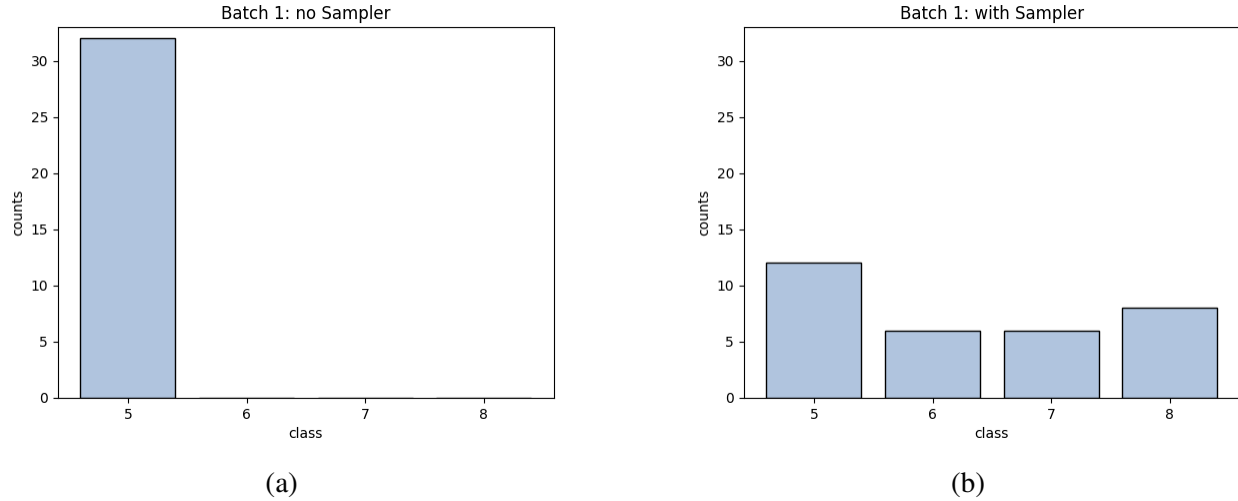


Figure 2.2: Class distribution without (a) and with (b) weighted sampler

2.3, where  $P_{i,j}$  is the probability that sample  $i$  belongs to class  $j$ . Since each experiment is multi-task, each task has its own loss function, and we minimize the weighted sum of these losses:  $\text{loss} = W_A \text{loss}_A + W_B \text{loss}_B$  for  $W_A = 2.0, W_B = 1.0$ , where  $A, B$  represent the task. Model parameters are optimized using the Adaptive Moment Estimation (Adam) algorithm.

$$\text{loss} = - \sum_{l=1}^L \sum_{k=1}^K y_{i,j} \log(P_{i,j}) \quad (2.3)$$

$$y_{i,j} = \begin{cases} 1, & \text{if class } j \text{ is correct} \\ 0, & \text{else} \end{cases} \quad (2.4)$$

### 2.3.1 Multi-Task Heads

Our models are trained on two tasks: (Task A) damage type discrimination and (Task B) damage severity discrimination. Task A is comprised of four classes, with 1 and 2 indicating mild and severe hysteretic damage and 3 and 4 indicating mild and severe earthquake-induced damage. Task A encourages the model to learn structural features that distinguish between fundamentally different sources of damage. This ensures that the model's embeddings capture damage-specific

patterns, rather than superficial characteristics of the training dataset.

Task B is comprised of four classes, with 5 indicating the undamaged state and 6, 7, and 8 indicating mild, moderate, and severe damage from bolt loosening. This teaches the model to learn a continuous spectrum of structural degradation, which is crucial when assessing real-world structures that exhibit progressive damage over time.

By learning both the type and severity of damage, the model develops rich, disentangled embeddings that separately capture what the damage is and how severe it is. This is critical for cross-domain evaluation with Z24, where the dataset contains progressive levels of different types of damage. The pretrained knowledge from Tasks A and B provides the model with a structured representation space, improving its ability to generalize onto unseen damage patterns.

### 2.3.2 Feature Space Analysis

The embedding features of the training, validation, and Z24 data are standardized onto a comparable scale. The scaling parameters are determined by the mean and standard deviation of the training embeddings, and applied to all of the embeddings. After scaling, a Linear Discriminant Analysis (LDA) model [8] is created with the number of components set to the smaller of: (1) the number of training classes minus one, or (2) the training embedding dimensionality. The LDA is then fit on the scaled training embeddings and corresponding class labels, learning a projection that maximally separates the classes in a lower-dimensional space. Classical LDA finds a transformation that maximizes the ratio of between-class scatter to within-class scatter. This fitted model is used to project, or transform, the validation and Z24 embeddings into the LDA-transformed feature space.

This LDA is extended by modeling the probabilistic structure of within-class and between-class variation using Probabilistic Linear Discriminant Analysis (PLDA) as outlined in [25]. PLDA assumes that an embedding can be decomposed into two terms:  $\mathbf{m}$ , which represents the mean, and  $A\mathbf{u}$ , which transforms projected class  $\mathbf{u}$  into the original data space by matrix  $A$ . Each embedding is represented as,

$$x = \mathbf{m} + A\mathbf{u} \quad (2.5)$$

$$\mathbf{u} \sim \mathcal{N}(\cdot | \mathbf{v}, \mathbf{I})$$

$$\mathbf{v} \sim \mathcal{N}(\cdot | 0, \Psi)$$

Where  $\mathbf{v}$  is the class in the original data space,  $\mathbf{I}$  is the identity covariance matrix for the within-class scatter, and  $\Psi$  is a diagonal covariance matrix for the between-class scatter. These embeddings are still high-dimensional; hence, to visualize the separability, the PLDA-transformed embeddings are reduced into two principal components using Principal Component Analysis (PCA) [8]. These visualizations will reveal whether PLDA has effectively captured class-specific structure by assessing the definition of clusters. We also quantify the definition using the overall silhouette score,

$$S = \frac{1}{N_e} \sum_{i=1}^{N_e} \frac{a_i - b_i}{\max(a_i, b_i)} \quad (2.6)$$

where  $N_e$  is the number of embeddings,  $a_i$  is the average intra-cluster distance, and  $b_i$  is average nearest-cluster distance.  $S \in [-1, 1]$ , so cluster definition is determined as follows:

- $S > 0.7$ : well-separated clusters
- $0.5 < S < 0.7$ : reasonably separated clusters
- $0.25 < S < 0.5$ : weak, overlapping clusters
- $S < 0.25$ : poor clustering

### 2.3.3 Likelihood-Based Scoring

Classification is performed in the latent space by scoring the similarity between the training embeddings (gallery) and the validation embeddings (probe). The probe  $\mathbf{u}^p$  is classified based on the log-likelihood ratio (LLR) [8], which represents the comparison between the probability that  $\mathbf{u}^p$  belongs to the same class as the  $n$  embeddings in the gallery vector  $\mathbf{u}^g$ , and the probability of

$\mathbf{u}^p$  and  $\mathbf{u}^g$  being independent. More explicitly, the log-likelihood ratio is

$$\begin{aligned} \log R(\mathbf{u}^p) &= \log \left( \frac{P(\mathbf{u}^p | \mathbf{u}^g)}{P(\mathbf{u}^p)P(\mathbf{u}^g)} \right) \\ &= \log P(\mathbf{u}^p | \mathbf{u}^g) - (\log P(\mathbf{u}^p) + \log P(\mathbf{u}^g)) \end{aligned} \quad (2.7)$$

The maximum LLR corresponds to the most likely class for a given probe from the validation embeddings and its associated classification score. Depending on the task, this can signify the damage case or the severity of the validation samples. In the context of hypothesis testing, the LLR quantifies the likelihood that the probe and gallery embeddings share the same latent class versus being independent. For validation data, where probes are assumed to belong to classes represented in the gallery, the LLR can be interpreted directly as a classification confidence, with higher values indicating stronger evidence that the probe belongs to a gallery class. The class corresponding to the maximum LLR is taken as the predicted class for the probe, which can then be compared to the true class to evaluate the model’s accuracy.

Taking all the LLRs, we can define target and non-target scores: targets are LLRs in which the probe and gallery belong to the same latent class, and non-targets are LLRs in which the probe and gallery do not belong to the same latent class. From these scores, we can compute standard classification metrics such as accuracy, precision, recall, and the F1 score, which quantify how well the model correctly identifies matching and non-matching pairs, balances false positives and false negatives, and provides an overall measure of verification performance.

## 2.4 Zero-Shot Evaluation

Evaluating the models on the Z24 data constitutes a zero-shot analysis, in which the class labels of the Z24 samples were not used during model training. Although the model has never seen these specific classes, the training samples were drawn from similar structural health monitoring experiments, so the learned representations and latent features remain relevant. As a result, the

model can still provide meaningful insights into the structural state or damage severity of the Z24 samples. This evaluation allows us to assess the model’s ability to generalize to unseen conditions and to test the robustness of its transfer learning capabilities.

This analysis is a binary verification, focusing on how well the models distinguish between damaged and undamaged states. To calculate the LLR, the gallery is the reference set of embeddings for a specific baseline class. Here, that baseline class is Class 0, the undamaged class. Rather than producing a predicted class as in the validation set, the LLR quantifies the similarity of each probe to the reference baseline, effectively measuring how much each probe deviates from known undamaged conditions. Projecting these embeddings into the same PCA space as the validation data allows for a qualitative comparison of latent space structure across seen and unseen datasets. Clusters that are farther apart indicate larger deviations in latent features, while well-defined clusters suggest that the model’s latent space generalizes effectively to new, unseen conditions.

The LLR analysis is divided into target and non-target scores to verify the model’s ability to distinguish undamaged from damaged samples. Target scores are computed via a leave-one-out approach, where each undamaged embedding is treated as a probe, and the remaining undamaged embeddings form the gallery. These scores quantify the likelihood that a sample belongs to the undamaged condition. Non-target scores are obtained by treating each damaged embedding as a probe and comparing it to the full set of undamaged embeddings as the gallery, quantifying the similarity between damaged probes and the undamaged baseline.

With the target and non-target scores, we can compute the False Positive Rate (FPR) and False Negative Rate (FNR) across a range of LLR thresholds. Equal Error Rate (EER) [8] is defined as the point at which FPR and FNR are closest to equal, and we report it as the average of FPR and FNR at this threshold. Additionally, the Area Under the ROC Curve [8] (AUC) can be computed directly from the LLRs and reflects the overall ability of the LLRs to discriminate between target and non-target pairs.

Next, we formalize a binary hypothesis test regarding the Z24 embeddings  $x_z$ :

- $H_0$ : The probe embedding and the gallery embeddings come from the same latent class

- $H_1$ : The probe embedding and the gallery embeddings come from different latent classes

The test statistic is the LLR between each damaged probe and the undamaged gallery (i.e., the non-target score). Each probe LLR is compared against a threshold  $\tau$  derived from the EER.

<p style="text-align: center;">If <math>\text{LLR} &gt; \tau \rightarrow \text{accept } H_0</math></p> <p style="text-align: center;">If <math>\text{LLR} \leq \tau \rightarrow \text{accept } H_1</math></p>
---

Rule 1: Decision rule for hypothesis testing based on the LLR and threshold  $\tau$ .

This procedure flags each probe as damaged or undamaged. From these decisions, we calculate binary classification accuracy, compute the confusion matrix, and analyze the AUC/ROC. This thresholded binary testing provides insights into the model’s ability to generalize and discriminate across new, unseen conditions.

For unsupervised cross-domain evaluation, the scaler, LDA, and PLDA are only trained using the experimental training data. Therefore, when testing on unseen data, the model relies solely on the representations learned from the source domain, without any exposure to the target domain statistics, which may affect generalization. For semi-supervised cross-domain evaluation, the scaler, LDA, and PLDA are trained using the experimental training data and the undamaged Z24 data. Therefore, when testing the unseen data, the model benefits from having seen target-domain undamaged samples, which can improve the alignment of latent representations and enhance its ability to discriminate between damaged and undamaged probes.

## Chapter 3: Results

In this study, we evaluate three experimental pipelines leveraging PLDA scoring to assess the MinGRU’s discriminative performance. Together, these experiments explore the quality and reliability of domain shift for damage identification in structural health monitoring. Our three methods of analysis are in-domain supervised evaluation, cross-domain supervised evaluation, and unsupervised cross-domain evaluation. The models are trained for 50 epochs with a hidden size of 128 and optimized with a learning rate of  $5e-4$ . Both models have a relatively low number of parameters, albeit the BiMinGRU having exactly double the number of parameters as the MinGRU as summarized in Table 3.1.

### 3.0.1 Supervised In-domain

To establish a baseline for how effectively the embeddings capture damage patterns, both the MinGRU and PLDA are trained and evaluated on experimental data. To visualize the PLDA-transformed embeddings, we use Principal Component Analysis (PCA) to project the embeddings in a two-dimensional space. PCA chooses the two principal components that have the most variance in the data. Figure 3.1 displays these projections, highlighting the clusters defined for each task.

For the MinGRU we see good separation in Task A classes and excellent separation in Task B classes. There is some confusion between classes in Task A, specifically between labels within their own damage class which more clearly seen in Figure 3.2. Highlighted in purple, we see a

Model	Number of Parameters
MinGRU	11914
BiMinGRU	23828

Table 3.1: Summary of Model Capacity

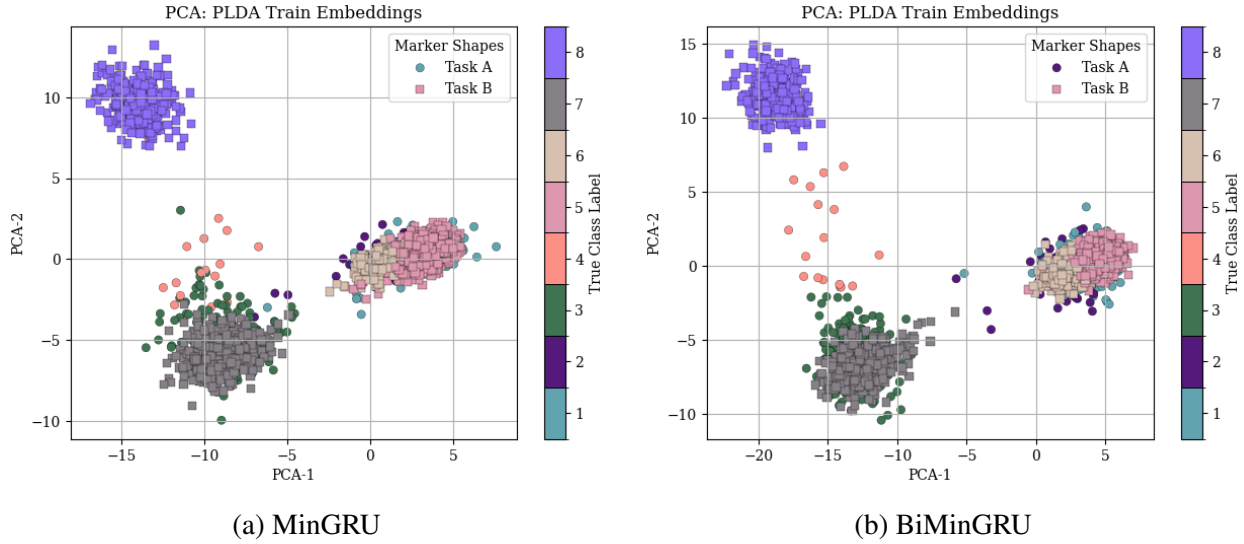


Figure 3.1: PCA projection of PLDA training embeddings for supervised in-domain setting

cluster of classes 1 and 2, which both represent Hysteretic damage. Highlighted in orange, we see a cluster of classes 3 and 4, which both represent Earthquake-induced damage. These two groupings indicate the model has learned to distinguish between the two types of damages well, but has trouble distinguishing the severity within these groupings. For Task B, we see two highly separated, defined clusters for the moderate and severe classes (7 and 8), and two clusters in close proximity for the mild and undamaged classes (5 and 6) in Figure 3.1. Despite the proximity of the undamaged and mildly damaged, these results demonstrate a strong discrimination between severity levels. The PLDA model achieved a silhouette score of 0.77, quantitatively supporting the visual observations that the embeddings form well-separated clusters.

For the BiMinGRU we see similar results, good separation in Task A classes and excellent separation in Task B classes. There is some confusion within Task A between labels belonging to the same damage class, similar to what was observed with the MinGRU. However, the two severity levels of earthquake-induced damage (3 and 4) are clearly separated, even though they remain in close proximity. Indicating, the bidirectional model has a greater ability to distinguish between the two types of damages and their severity. For Task B, we again see two highly separated, defined clusters for the moderate and severe classes, and two clusters in close proximity for the mild and undamaged classes, demonstrating strong discrimination between severity levels, as seen in the

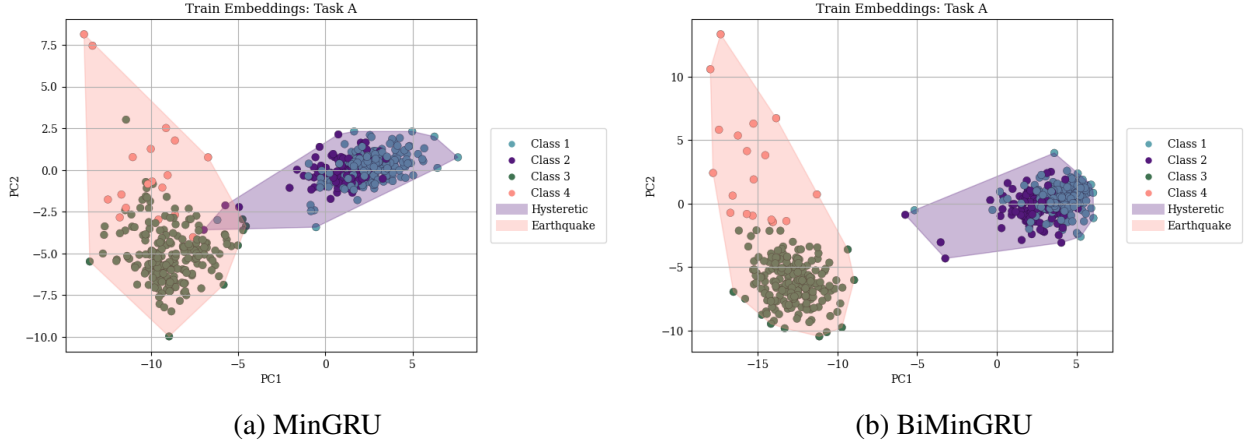


Figure 3.2: PCA projection of PLDA Task A training embeddings for supervised in-domain setting MinGRU. However, due to the improvements in separation seen in Task A, the BiMinGRU model appears to outperform the MinGRU overall. This is confirmed by the silhouette score achieved by the PLDA model, 0.80, which corroborates the visual observations and suggests a modest performance advantage over the MinGRU.

To verify the model’s classification abilities, we test the PLDA model on a test set from the same experimental datasets. Based on the overlap between clusters seen in Figure 3.1, we expect most misclassifications to occur between labels of the same damage type in Task A for both models. Figure 3.3 summarizes the classification results and highlights the discrepancy in class sizes between each task. For the MinGRU model, as predicted, most errors occur in Task A and are between labels within the same damage type. However, there are a few exceptions, where 3 Hysteretic damage embeddings were classified as Earthquake-induced and 1 mild Bolt Loosened embedding was classified as undamaged. The BiMinGRU model only has errors within the damage type classes of Task A, and otherwise has perfect classification. The misclassifications for each model for Task A embeddings can be further highlighted in Figure 3.4. There is an apparent accuracy difference between the two tasks, which is likely due to two factors: discrepancy in task size and the progressive nature of the experiments. The task with more embeddings contains greater variability, which may allow the model to learn a wider range of damage patterns more effectively. Additionally, because each experiment was performed progressively, it is reasonable to accept that

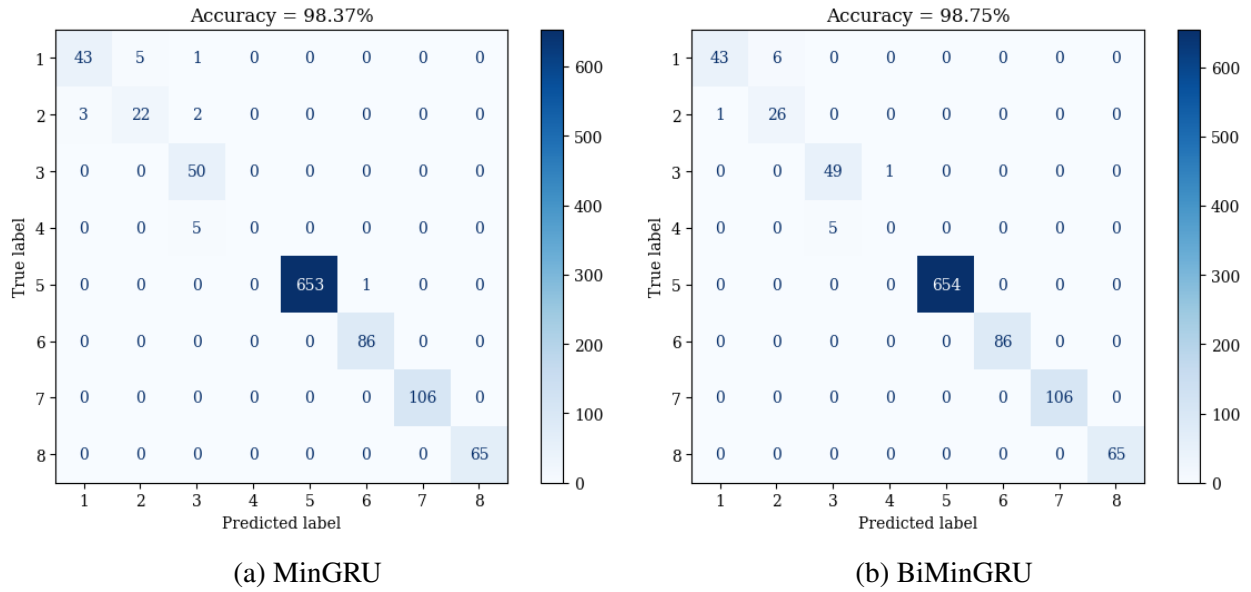


Figure 3.3: Confusion matrix summarizing in-domain classification results

most errors occur between adjacent levels of damage.

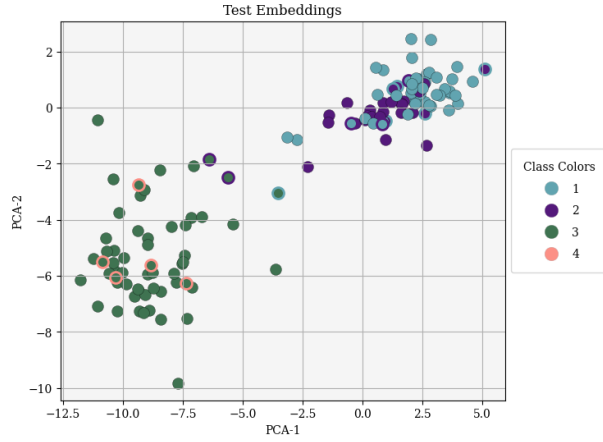
As summarized in Table 3.2, all models perform well at evaluating in-domain data, with some expected classification errors. The BiMinGRU model outperformed the MinGRU, particularly in Task A, indicating that bidirectional processing enhances the quality of learned embeddings for in-domain sequential data.

	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)
MinGRU	98.37	97.98	98.37	98.14
BiMinGRU	98.75	98.48	98.75	98.58

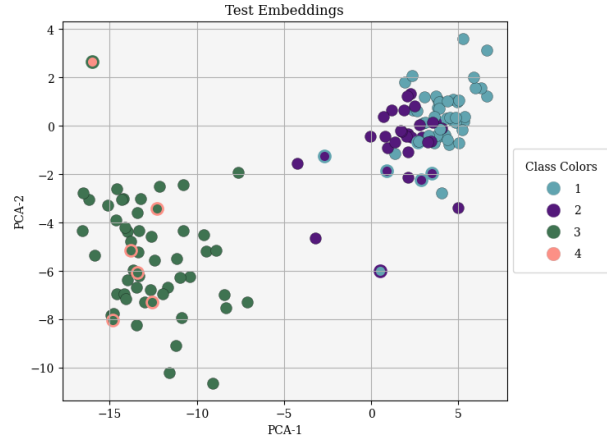
Table 3.2: Performance metrics from supervised in-domain classification

### 3.0.2 Unsupervised Cross-domain

To evaluate generalization to unseen structures under domain shift in an unsupervised setting, the PLDA model trained on experimental data in the prior experiment is applied to the Z24 dataset. Figure 3.5 presents the projected embeddings for both models, where substantial overlap between damaged and undamaged states is observed. Consistent with this visual assessment, the silhouette scores are negative for both models, with values of  $-0.1029$  for the MinGRU and  $-0.1103$  for the

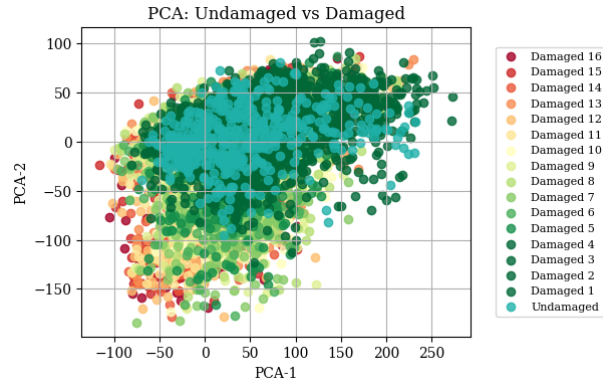


(a) MinGRU

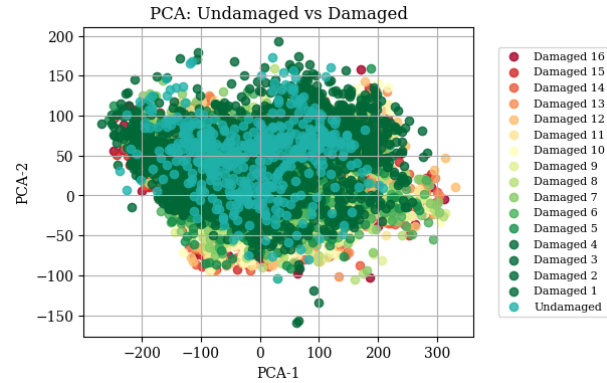


(b) BiMinGRU

Figure 3.4: Task A Misclassifications: The edge color denotes the true class label, whereas the fill color denotes the predicted class label.



(a) MinGRU



(b) BiMinGRU

Figure 3.5: PCA projection of PLDA Z24 test embeddings for unsupervised cross-domain setting

BiMinGRU, indicating poor overall separability in this setting.

To further examine separability at the extremes of damage severity, Figure 3.6 restricts the visualization to the undamaged state and the most severely damaged condition. In this case, a modest improvement in separation is observed for the MinGRU relative to the BiMinGRU. This trend is reflected in the silhouette scores, which increase to 0.0287 for the MinGRU and 0.0268 for the BiMinGRU.

Model classification performance is assessed by predicting the structural state of Z24 using LLR scores, which indicate whether an embedding is more consistent with damaged or undamaged

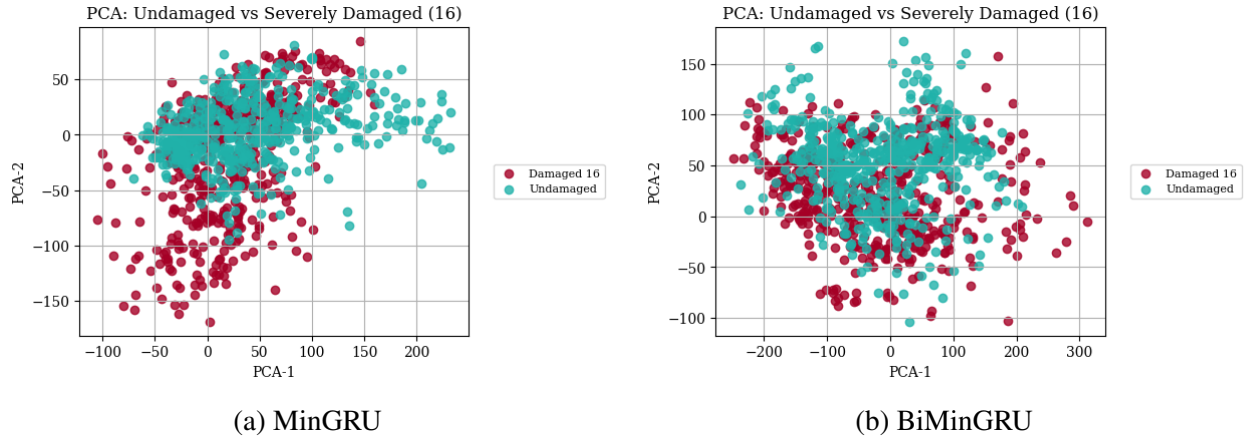


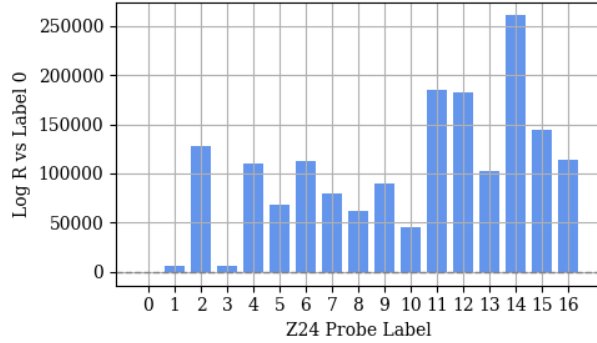
Figure 3.6: PCA projection of PLDA Z24 extreme embeddings for unsupervised cross-domain setting

behavior. The Z24 dataset has one undamaged state (Label 0) and 16 damaged states (Labels 1-16). The undamaged state is used as the reference baseline, and LLR scores are computed to quantify the statistical separation between each damaged state and this baseline, enabling a binary classification of damaged versus undamaged. Larger LLR values indicate greater deviation from the undamaged condition.

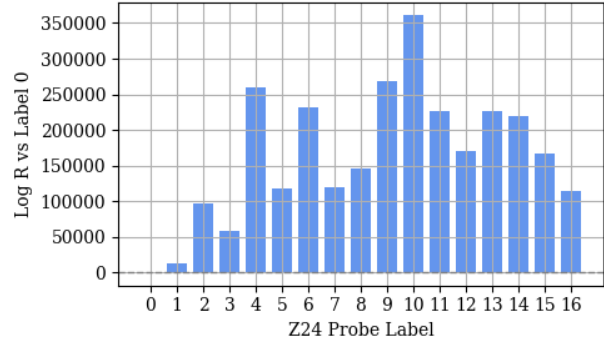
Figure 3.7 displays the LLR scores across damage states for both models. For the MinGRU, LLR scores generally increase with damage severity, consistent with expectations for progressive structural degradation. However, notable deviations are observed: specifically, Label 2 exhibits a substantially higher LLR score than its adjacent labels, and Label 10 displays a far lower LLR score than expected given its relative severity. The BiMinGRU scores also a similar increasing trend at lower damage levels, indicating increased sensitivity to early damage progression. However, it does not consistently assign the highest LLR scores to the most severe damage states (Labels 13–16), suggesting reduced discrimination among later stages of damage severity.

Following the zero-shot framework, the decision threshold is chosen at the point where the false positive and false negative rates are approximately equal, yielding the EER as a single, interpretable measure of the trade-off between false alarms and missed detections. This threshold is then used to perform hypothesis testing on the log-likelihood ratio scores, producing binary damage decisions.

The MinGRU achieves an EER of 6.44% and an AUC of 0.991, while the BiMinGRU attains



(a) MinGRU



(b) BiMinGRU

Figure 3.7: Bar graph of log-likelihood ratio scores in unsupervised cross-domain setting

a slightly lower EER of 6.24% and a higher AUC of 0.994. Applying Rule 1, all embeddings classified as undamaged are assigned Label 0, while damaged embeddings are assigned Label 1. These predictions are compared with the ground-truth Z24 labels, where Label 0 denotes the undamaged state, and Labels 1–16 are grouped as damaged and reassigned to Label 1. From this comparison, accuracy and F1 scores are computed for each model and reported in Table 3.3, along with the corresponding confusion matrices in Figure 3.8 for visualization.

	Accuracy (%)	F1 (%)	EER (%)	AUC
MinGRU	93.72	63.56	6.44	0.991
BiMinGRU	93.74	63.74	6.24	0.994

Table 3.3: Performance metrics from unsupervised cross-domain classification

The resulting accuracies and F1 scores suggest moderate classification performance for both models. The BiMinGRU achieves marginally higher accuracy and F1 score than the MinGRU; however, the observed differences are very small, on the order of a few hundredths of a percent. As shown in the confusion matrices, both models misclassify 497 damaged embeddings as undamaged. Notably, all of these misclassified samples correspond to Label 1 in the original Z24 labeling, which represents the lowest severity damage state associated with the installation of the pier settlement system.

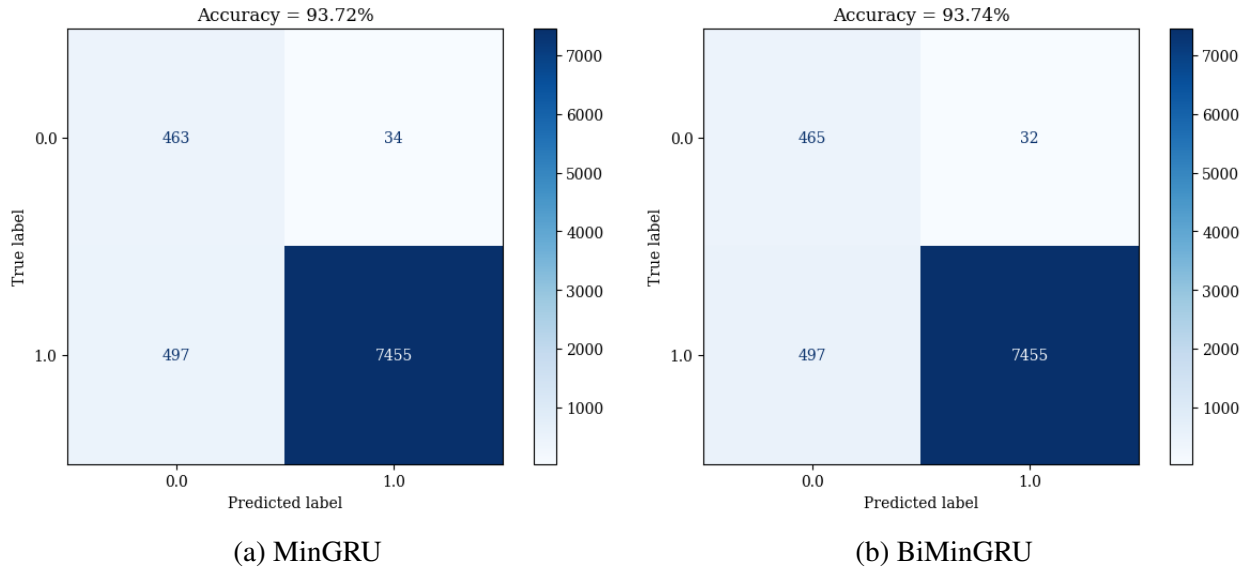


Figure 3.8: Confusion matrix summarizing unsupervised cross-domain classification results

### 3.0.3 Supervised Cross-domain

To evaluate the transferability of embeddings to in-service structures under domain shift, the models are trained on experimental data, while the PLDA is trained on both experimental and Z24 undamaged data. Undamaged data is used here for training under the assumption that in practical deployment of this strategy, there would be access to data of the structure early on in its operation, before damage could realistically occur. Damage state testing is performed using the Z24 damaged data.

Figure 3.9 presents the projected embeddings for both models, illustrating overlap between undamaged and damaged states. Relative to the unsupervised cross-domain setting, the overlap is reduced, suggesting improved separability. However, the silhouette scores remain negative, with values of  $-0.1917$  for the MinGRU and  $-0.2299$  for the BiMinGRU, indicating that overall geometric separability remains limited when considering all damaged states collectively.

For consistency, the analysis is further restricted to the undamaged state and the most severely damaged condition in Figure 3.10. In this simplified comparison, clearer separation is observed than in the unsupervised cross-domain case, with the BiMinGRU exhibiting more pronounced sep-

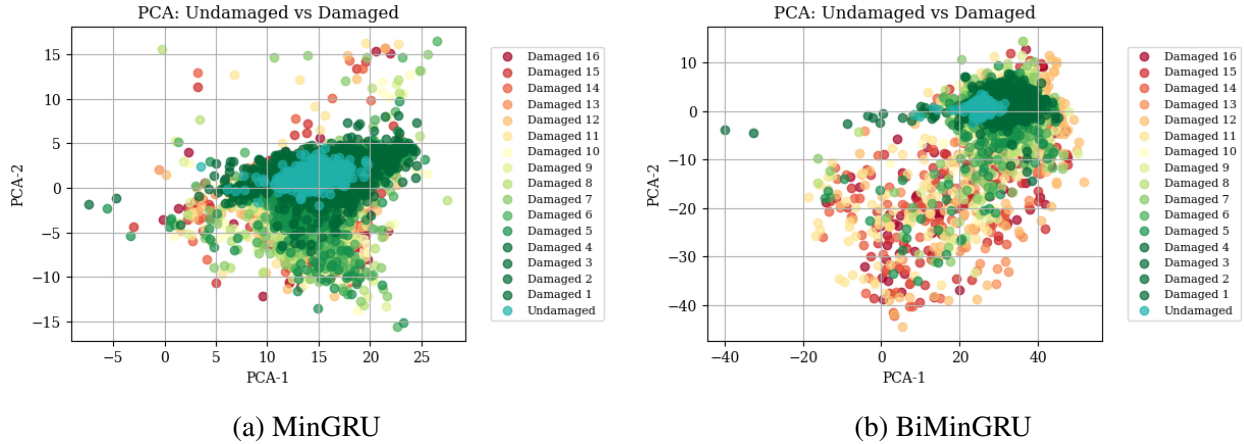


Figure 3.9: PCA projection of PLDA Z24 test embeddings for supervised cross-domain setting

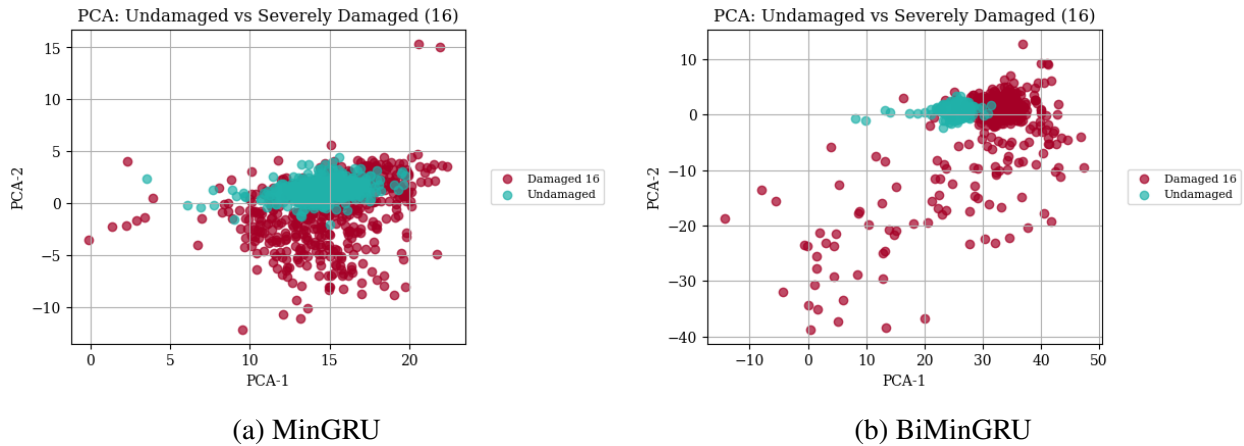


Figure 3.10: PCA projection of PLDA Z24 extreme embeddings for supervised cross-domain setting

eration than the MinGRU. This trend is supported by the corresponding silhouette scores, which increase to 0.0220 for the MinGRU and 0.2051 for the BiMinGRU, indicating substantially stronger separability for the BiMinGRU in this extreme-damage comparison.

Moving on to classification, we repeat the zero-shot framework as in the previous section. Although the PLDA model is trained using embeddings from the undamaged condition, and therefore does not constitute a strictly zero-shot setting, this evaluation assesses generalization to unseen damaged states and therefore serves as a meaningful test of damage detection capability under limited supervision.

The MinGRU achieves an EER of 0.102% and an AUC of 0.998, while the BiMinGRU achieves

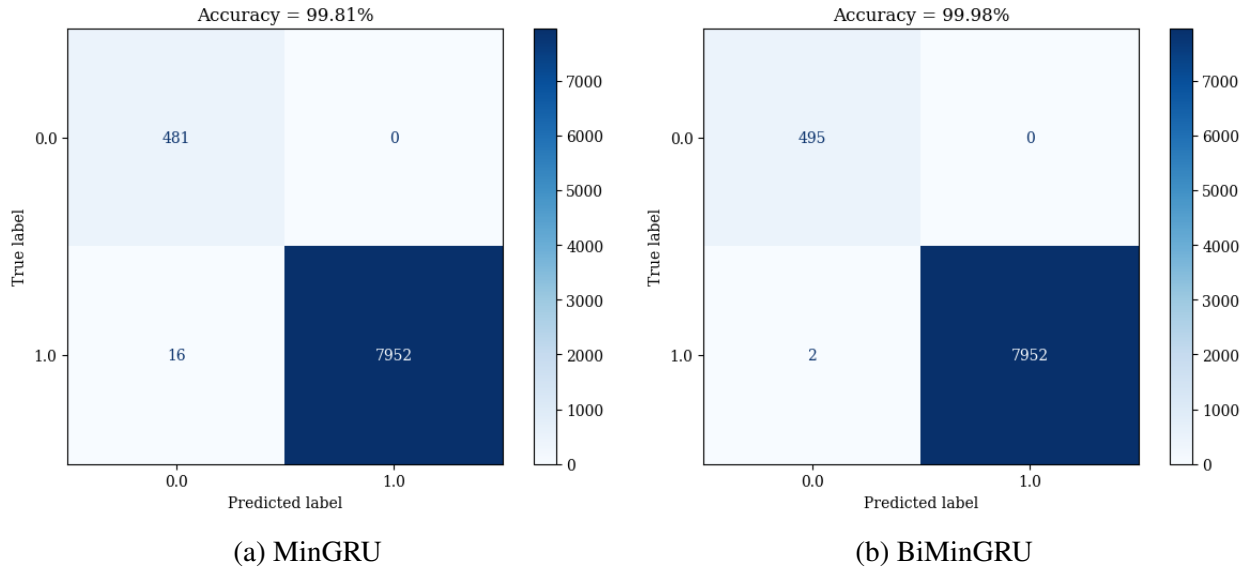


Figure 3.11: Confusion matrix summarizing supervised cross-domain classification results

a slightly lower EER of 0.101% and slightly higher AUC of 0.999. Applying Rule 1, binary damage predictions are obtained and compared against the ground-truth Z24 labels. The resulting classification metrics are summarized in 3.4. The MinGRU achieves an accuracy of 99.81% and an F1 of 99.74, while the BiMinGRU achieves an accuracy of 99.99% and an F1 of 99.90. Figure 3.11 presents the corresponding confusion matrices, showing near-perfect classification performance with 16 missed detections for the MinGRU and 2 missed detections for the BiMinGRU.

	Accuracy (%)	F1 (%)	EER (%)	AUC
MinGRU	99.81	99.74	0.102	0.998
BiMinGRU	99.99	99.90	0.101	0.999

Table 3.4: Performance metrics from supervised cross-domain classification

## Chapter 4: Discussion

This chapter evaluates damage detection and generalization performance using MinGRU and BiMinGRU embeddings under supervised and unsupervised settings. Across experiments, both models demonstrated strong damage detection capability, with differences emerging in early-stage sensitivity, severity discrimination, and cross-domain behavior.

### 4.1 Embedding Separability: Metrics and Visualization

Across the different evaluation settings, the learned embeddings exhibit markedly different separability characteristics, highlighting the strong influence of supervision and domain alignment on representation quality. In the supervised in-domain setting, embeddings form well-defined clusters corresponding to damage categories and severity levels. Meanwhile, separability reduces substantially under the Z24 domain shift, particularly in the fully unsupervised cross-domain case. These trends are consistent across both visualizations and quantitative separability metrics, though the degree of alignment between the two varies by setting.

In the supervised in-domain experiment, the strong separability observed across tasks indicates that the models effectively capture damage-related structure when trained and evaluated on data drawn from the same domain. In contrast, both supervised and unsupervised cross-domain settings exhibit increased overlap between undamaged and damaged states, reflecting the challenges posed by fundamental differences between the experimental data and the Z24 dataset.

The comparison between visual separation methods and silhouette scores highlights an important distinction between what appears separated by eye and what is measured quantitatively. In several cases, projected embeddings appear visually better separated than others despite silhouette scores suggesting otherwise. This apparent discrepancy arises because silhouette scores are

sensitive to within-class dispersion and boundary overlap, whereas low-dimensional visualizations emphasize the main separation directions. As a result, visually distinct clusters do not necessarily correspond to strong overall separability as measured by distance-based metrics.

Conversely, modest improvements in silhouette score for reduced comparisons, such as undamaged versus severely damaged, confirm that separability is highly contextual. These findings underscore the importance of interpreting embedding visualizations and separability metrics as complementary tools rather than interchangeable.

## 4.2 Generalization Under Domain Shift

Through these experiments we observe a clear pattern between model performance, supervision, domain alignment, and damage severity. In the supervised in-domain, both models achieve strong classification performance, with most errors occurring between closely related damage classes. These errors are largely confined to Task A, where labels represent different severity levels within the same damage mechanism. This behavior is expected, given the progressive nature of the experiments and the physical similarity between adjacent damage states.

When generalization is evaluated under unsupervised cross-domain conditions, classification performance reduces substantially. Although both models retain the ability to distinguish damaged from undamaged states, errors increase markedly, particularly for low-severity damage. Nearly all misclassifications correspond to the lowest damage state in the Z24 dataset, indicating that early-stage damage remains difficult to detect when transferring from experimental data to an unseen structure. Despite this challenge, the models maintain moderate binary performance, with comparable EER and AUC values, demonstrating that likelihood-based classification remains viable even under significant domain shift.

The supervised cross-domain setting shows a dramatic improvement in classification performance compared to the unsupervised cross-domain. Training the PLDA model using undamaged embeddings from the target domain enables near-perfect discrimination between damaged and undamaged states, with only a small number of missed detections. This result highlights the sig-

nificant influence having even limited domain-specific information can have for improving generalization.

Taken together, these results demonstrate that classification generalization depends strongly on both domain alignment and damage severity. While both models perform well in-domain and under limited supervision, unsupervised cross-domain detection remains challenging for low-severity damage. At the same time, the strong performance observed for the supervised cross-domain experiment shows the potential of embedding-based, likelihood-driven approaches for practical damage detection scenarios where labeled data are scarce but not entirely absent.

### 4.3 Comparison of MinGRU and BiMinGRU

Across all experiments, the MinGRU and BiMinGRU models exhibit broadly similar behavior, with only modest and context-dependent performance differences. While the BiMinGRU consistently matches or slightly outperforms the MinGRU in several settings, these improvements are generally small, suggesting that both architectures learn comparably effective patterns for damage detection.

In terms of embedding separability, both models produce well-structured representations in the supervised in-domain setting, with clear clustering by damage type and severity. The BiMinGRU shows slightly cleaner separation in certain cases, particularly when distinguishing extreme damage states under supervised cross-domain conditions. However, these gains are not uniform across experiments. In the unsupervised cross-domain setting, differences between the two models are minimal, and both exhibit weak overall separability when all damaged states are considered collectively. This indicates that the relative impact of domain shift is greater than architectural differences.

Classification results follow a similar pattern. In the supervised in-domain setting, both models achieve high accuracy, with most misclassifications occurring between adjacent damage states. The BiMinGRU produces fewer errors than the MinGRU, particularly in Task A, suggesting that bidirectional temporal context may help resolve subtle distinctions when sufficient labeled data

are available. In the unsupervised cross-domain setting, however, both models experience a noticeable drop in performance, especially for low-severity damage. This indicates the differences between architectures become negligible under these conditions. Under supervised cross-domain conditions, classification performance improves substantially for both models compared to the unsupervised, with the BiMinGRU again achieving slightly fewer missed detections.

An important consideration in interpreting these results is model capacity. The BiMinGRU contains approximately twice the number of parameters as the MinGRU, raising the question of whether observed performance gains can be attributed to bidirectionality itself or simply to increased capacity. The BiMinGRU’s modest advantages may be partially attributed to its bidirectional structure, which could help reduce the influence of short-term noise by incorporating additional temporal context. However, the observed performance differences are small and inconsistent across experiments, and do not provide strong evidence that bidirectionality or increased parameter count alone leads to substantial improvements.

Overall, the results suggest that bidirectional processing offers only incremental benefits. The MinGRU achieves competitive performance with fewer parameters, indicating that much of the relevant damage-related information can be captured using a lightweight, unidirectional architecture. From a practical perspective, this finding highlights a favorable trade-off between model complexity and performance: the MinGRU provides strong baseline performance at lower computational cost.

## Chapter 5: Conclusion

Early damage detection remains a central challenge in structural health monitoring, particularly when labeled data from damaged, in-service structures are scarce or unavailable. While vibration-based methods offer a promising avenue for continuous monitoring, their effectiveness depends critically on the ability to generalize across structural configurations, environmental conditions, and damage scenarios. This thesis investigated whether the Minimal Gated Recurrent Unit (MinGRU) neural network embeddings could support robust damage detection under such constraints.

To this end, cepstral coefficient features extracted from vibration measurements were combined with MinGRU and BiMinGRU architectures to learn compact representations of structural response. These embeddings were evaluated using probabilistic linear discriminant analysis across supervised, unsupervised, in-domain, and cross-domain settings. The results demonstrate that both architectures learn highly informative representations when training and evaluation occur within the same domain, yielding strong embedding separability and near-perfect classification performance. In this setting, misclassifications primarily occur between closely related damage states, reflecting the physical similarity of adjacent severity levels rather than significant failure to distinguish damaged from undamaged behavior.

When evaluated under domain shift using the Z24 bridge dataset, performance reduces substantially, particularly for low-severity damage. Both embedding separability and classification accuracy are reduced in the fully unsupervised cross-domain setting, highlighting the sensitivity of vibration-based damage detection to domain alignment. Nevertheless, the results show that severe damage remains distinguishable even without supervision, indicating that extreme deviations in structural response generalize more reliably across domains. Importantly, introducing limited supervision by training on undamaged data from the Z24 dataset leads to a dramatic improvement in classification performance, underscoring the value of even minimal domain-specific information

in practical SHM applications.

Across all experiments, differences between the MinGRU and BiMinGRU architectures were modest and inconsistent. While the BiMinGRU occasionally exhibited slightly improved separability or reduced misclassification in specific settings, these gains were small relative to the increase in model complexity. Overall, the MinGRU achieved competitive performance with fewer parameters and reduced computational cost, suggesting that lightweight sequence models are sufficient to capture much of the damage-related information present in vibration data. These findings indicate that architectural simplicity can be a practical advantage in long-term monitoring scenarios where efficiency and scalability are critical.

In summary, this work demonstrates that embedding-based, likelihood-driven approaches can support effective damage detection under limited supervision, even in the presence of domain shift. At the same time, the results highlight persistent challenges in detecting early-stage damage and generalizing across structurally dissimilar systems. Addressing these challenges through improved broader experimental coverage and improved handling of class imbalance represents an important direction for future research. Taken together, the findings of this thesis contribute to the development of practical, data-efficient methods for structural health monitoring in real-world settings.

## References

- [1] K. Worden and G. Manson, “The application of machine learning to structural health monitoring,” *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 365, no. 1851, pp. 515–537, Dec. 2006. eprint: <https://royalsocietypublishing.org/rsta/article-pdf/365/1851/515/318994/rsta.2006.1938.pdf>.
- [2] A. Rytter, *Vibrational based inspection of civil engineering structures*, Ph.D.-Thesis defended publicly at the University of Aalborg, April 20, 1993 PDF for print: 206 pp., Denmark, 1993.
- [3] C. R. Farrar, S. W. Doebling, and D. A. Nix, “Vibration-based structural damage identification,” *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 359, no. 1778, 131–149, 2001.
- [4] H. Melhem and H. Kim, “Damage detection in concrete by fourier and wavelet analyses,” *Journal of Engineering Mechanics*, vol. 129, no. 5, pp. 571–577, 2003.
- [5] I. Yesilyurt and H. GURSOY, “Estimation of elastic and modal parameters in composites using vibration analysis,” *Journal of Vibration and Control*, vol. 21, no. 3, pp. 509–524, 2015. eprint: <https://doi.org/10.1177/1077546313486275>.
- [6] C. Smith, C. M. Akujuobi, P. Hamory, and K. Kloesel, “An approach to vibration analysis using wavelets in an application of aircraft health monitoring,” *Mechanical Systems and Signal Processing*, vol. 21, no. 3, pp. 1255–1272, 2007.
- [7] N. Cheraghi, G. P. Zou, and F. Taheri, “Piezoelectric-based degradation assessment of a pipe using fourier and wavelet analyses,” *Computer-Aided Civil and Infrastructure Engineering*, vol. 20, no. 5, 369–382, 2005.
- [8] H. Beigi, *Fundamentals of Speaker Recognition*. New York: Springer, 2011, ISBN: 978-0-387-77591-3.
- [9] L. Balsamo, R. Betti, and H. Beigi, “A structural health monitoring strategy using cepstral features,” *Journal of Sound and Vibration*, vol. 333, no. 19, pp. 4526–4542, 2014.
- [10] C. Wang et al., “An lstm-based detection model for breathing cracks with multiple damage positions and degrees of beam-like bridges under vehicle loads,” *Structural Health Monitoring*, vol. 0, no. 0, p. 14759217251319592, 0. eprint: <https://doi.org/10.1177/14759217251319592>.

- [11] M. Azimi, A. D. Eslamlou, and G. Pekcan, “Data-driven structural health monitoring and damage detection through deep learning: State-of-the-art review,” *Sensors*, vol. 20, no. 10, 2020.
- [12] E. M. Tronci, H. Beigi, R. Betti, and M. Q. Feng, “A damage assessment methodology for structural systems using transfer learning from the audio domain,” *Mechanical Systems and Signal Processing*, vol. 195, p. 110 286, 2023.
- [13] E. Reynders and G. D. Roeck, “Continuous vibration monitoring and progressive damage testing on the z24 bridge,” *Encyclopedia of Structural Health Monitoring*, 2008.
- [14] L. Feng, F. Tung, M. O. Ahmed, Y. Bengio, and H. Hajimirsadeghi, *Were rnns all we needed?* 2024. arXiv: 2410.01201 [cs.LG].
- [15] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [16] A. Graves, “Long short-term memory,” in *Supervised Sequence Labelling with Recurrent Neural Networks*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 37–45, ISBN: 978-3-642-24797-2.
- [17] K. Cho et al., *Learning phrase representations using rnn encoder-decoder for statistical machine translation*, 2014. arXiv: 1406.1078 [cs.CL].
- [18] L. Feng, F. Tung, and H. Hajimirsadeghi, *Minimal lstms and grus: Simple, efficient, and fully parallelizable*, <https://rbcboREALIS.com/research-blogs/minimal-lstms-and-grus-simple-efficient-and-fully-parallelizable/>, 2025.
- [19] A. Vaswani et al., *Attention is all you need*, 2023. arXiv: 1706.03762 [cs.CL].
- [20] B. P. Bogert, “The quefreny analysis of time series for echoes : Cepstrum, pseudo-autocovariance, cross-cepstrum and saphe cracking,” 1963.
- [21] D. Mitrović, M. Zeppelzauer, and C. Breiteneder, “Chapter 3 - features for content-based audio retrieval,” in *Advances in Computers: Improving the Web*, ser. Advances in Computers, vol. 78, Elsevier, 2010, pp. 71–150.
- [22] E. Figueiredo, G. Park, J. Figueiras, C. Farrar, and K. Worden, “Structural health monitoring algorithm comparisons using standard data sets,” *Encyclopedia of structural health monitoring*, 2009.
- [23] Y. L. Mo, G. Song, Y. Moslehy, H. Gu, D. H. Sangers, and D. Belarbi, *Nesr payload: Damage detection of reinforced concrete columns subjected to com- bined actions*.

- [24] J. MAECK and G. DE ROECK, “Description of z24 benchmark,” *Mechanical Systems and Signal Processing*, vol. 17, no. 1, 127–131, 2003.
- [25] S. Ioffe, “Probabilistic linear discriminant analysis,” in *Computer Vision – ECCV 2006*, A. Leonardis, H. Bischof, and A. Pinz, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 531–542, ISBN: 978-3-540-33839-0.