

Uncertainty-Aware Neural Networks for Engineering Risk Assessment
and Decision Support

by

Rahul Rathnakumar

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved April 2024 by the
Graduate Supervisory Committee:

Yongming Liu, Chair
Hao Yan
Suren Jayasuriya
Houlong Zhuang
Beomjin Kwon

ARIZONA STATE UNIVERSITY

May 2024

©2024 Rahul Rathnakumar

All Rights Reserved

ABSTRACT

This dissertation contributes to uncertainty-aware neural networks using multi-modality data, with a focus on industrial and aviation applications. Drawing from seminal works in recent years that have significantly advanced the field, this dissertation develops techniques for incorporating uncertainty estimation and leveraging multi-modality information into neural networks for tasks such as fault detection and environmental perception. The escalating complexity of data in engineering contexts demands models that predict accurately and quantify uncertainty in these predictions. The methods proposed in this document utilize various techniques, including Bayesian Deep Learning, multi-task regularization and feature fusion, and efficient use of unlabeled data. Popular methods of uncertainty quantification are analyzed empirically to derive important insights on their use in real world engineering problems. The primary objective is to develop and refine Bayesian neural network models for enhanced predictive accuracy and decision support in engineering. This involves exploring novel architectures, regularization methods, and data fusion techniques. Significant attention is given to data handling challenges in deep learning, particularly in the context of quality inspection systems. The research integrates deep learning with vision systems for engineering risk assessment and decision support tasks, and introduces two novel benchmark datasets designed for semantic segmentation and classification tasks. Additionally, the dissertation delves into RGB-Depth data fusion for pipeline defect detection and the use of semi-supervised learning algorithms for manufacturing inspection tasks with imaging data. The dissertation contributes to bridging the gap between advanced statistical methods and practical engineering applications.

DEDICATION

In memory of Mahadevan Thatha.

ACKNOWLEDGMENTS

I would like to acknowledge a number of people whose support was invaluable in the completion of this PhD thesis.

First and foremost, my heartfelt thanks to Dr. Yongming Liu, my supervisor, for his invaluable insights, patience, and unwavering support throughout this journey. Your dynamic leadership in research is not just admirable but something I aspire to emulate.

Immense gratitude is also due to Dr. Hao Yan and Dr. Suren Jayasuriya. Your guidance in research group meetings and coursework significantly contributed to the progress of this dissertation. I'm equally thankful to Dr. Houlong Zhuang and Dr. Beomjin Kwon for their insightful suggestions and constructive feedback, which were instrumental in enhancing the quality of this work.

I'm profoundly grateful to my colleagues Yutian Pang, Hari Iyer, and Jiayu Huang for the many hours of fruitful collaboration and stimulating discussions. My appreciation also extends to Gowtham, Abhishek, Sampriti, Rakesh, and Rohit for being great teammates during my time at ASU.

To the most special people in my life – my parents, Rathnakumar and Rohini, thank you for your immeasurable sacrifices and love, which made this journey possible. My sister, Priya, whose belief in me often surpassed my own. My partner, Dhun, who has been a constant pillar of support and love, especially during the most challenging times, for which I am eternally grateful.

Finally, I acknowledge the Mechanical and Aerospace Engineering Department at ASU for their support in my dissertation work. My gratitude also extends to US-DOT and NASA for funding two significant projects that greatly enhanced my skills as an engineer and researcher.

TABLE OF CONTENTS

	Page
LIST OF TABLES	x
LIST OF FIGURES	xii
CHAPTER	
1 INTRODUCTION	1
1.1 Overview	1
1.1.1 Data Handling	2
1.1.2 Data Fusion	3
1.1.3 Semi-supervised Learning in Industrial Informatics	5
1.1.4 Uncertainty Quantification	6
1.2 Outline of the Dissertation	6
2 BACKGROUND	10
2.1 Neural Network Models	10
2.1.1 Inductive Biases in Neural Networks	11
2.2 Uncertainty Quantification	13
2.3 Bayesian Neural Networks	15
2.3.1 Bayesian Parameterization of Neural Networks	15
2.3.2 Dropout Variational Inference	17
3 BAYESIAN ENTROPY NEURAL NETWORKS FOR PHYSICS-AWARE PREDICTION	19
3.1 Introduction	19
3.2 Method	22
3.2.1 Bayesian Entropy Method for Incorporating Constraint Information	22

CHAPTER	Page
3.2.2 Bayesian Entropy Neural Networks	24
3.3 Experiments	26
3.3.1 1D-Regression	26
3.3.1.1 Incorporating Value and Derivative Constraints to the Prediction	27
3.3.1.1.1 Value Constraints Outside Training Range.	28
3.3.1.1.2 Variance constraints	28
3.3.1.1.3 Conflicting Constraint and Variance Constraint at Same Location.	29
3.3.1.1.4 Bound Constraints	29
3.3.2 Beam Deflection Problem	31
3.3.2.0.1 Constraint Implementation	33
3.3.3 Microstructure Generation	35
3.3.3.1 Bayesian Entropy Convolutional Variational Auto- Encoder (BE-CVAE)	36
3.3.3.2 Dataset	36
3.3.3.3 Constraint - Two Point Correlation Function	37
3.3.3.4 Constraint - Porosity	39
3.3.3.5 Results	39
3.4 Conclusion	41
4 EPISTEMIC AND ALEATORIC UNCERTAINTY QUANTIFICA- TION FOR CRACK DETECTION USING A BAYESIAN BOUNDARY AWARE CONVOLUTIONAL NETWORK	44
4.1 Introduction	44

CHAPTER	Page
5.3.4 Post-processing and Defect Measurement	96
5.4 Experimental Setup	98
5.4.1 Dataset	98
5.4.2 Implementation Details	99
5.5 Results and Discussion	99
5.6 Conclusions	103
6 DEFECT SEGMENTATION WITH LIMITED LABELED DATA USING CONSISTENCY REGULARIZATION AND ACTIVATION MAP INTERPOLATION	105
6.1 Introduction	105
6.2 Related Work	106
6.2.1 Image Segmentation for Defect Detection	106
6.2.2 Semi-Supervised Segmentation	108
6.3 Methodology	111
6.3.1 Overview of the Proposed Framework	112
6.3.2 Loss Functions	115
6.3.3 Dataset Batching	116
6.4 Experiments	117
6.4.1 Network Setup and Training	118
6.4.2 Prediction	120
6.5 Results and Discussion	122
6.6 Conclusion	127
7 UNCERTAINTY-AWARE DECISION SUPPORT USING ON-BOARD VISION DATA FOR PILOT SITUATIONAL AWARENESS	128

CHAPTER	Page
7.1 Introduction	128
7.2 Related Work	131
7.2.1 Decision Support Systems in Aviation	131
7.2.2 Threat Prediction Models in General Aviation	132
7.3 Methodology	133
7.3.1 Overview	133
7.3.2 Problem Setup	134
7.3.2.1 Weather Threats Prediction	134
7.3.2.2 Cloud Cover Classification	135
7.3.2.3 Instrument Meteorological Condition Prediction	136
7.3.3 Multilabel, Multiclass Classification Model for Threat Pre- diction.....	137
7.3.3.1 Image Encoder Module	137
7.3.3.2 Graph Representation Learning Module	138
7.3.3.3 Sentence Transformer Module	141
7.3.3.4 Multitask Supervised Learning	141
7.4 Experimental Setup	142
7.4.1 GA-SA Dataset	142
7.4.1.1 Weather Description using METAR and Phrase Em- beddings	145
7.4.2 Training and Evaluation Setup	146
7.4.3 Evaluation Metrics	147
7.5 Results and Discussion.....	148
7.5.1 Model Evaluation	148

CHAPTER	Page
7.5.2 Discussion and Limitations	150
7.6 Conclusion	153
8 CONCLUSION	155
REFERENCES	159

LIST OF TABLES

Table	Page
1. Comparison of BENN and PR-BNN Methods for Value, Bound and Derivative Constraints	31
2. Constraint Compliance as TPCF-L1 Error on Generated Samples: Analysis along Training Data and Number of Constraints	41
3. Architecture Details of the Segmentation Network	62
4. Training Details and Hyperparameters	63
5. Performance Comparison for Models Trained on DeepCrack and CrackForest Dataset	66
6. Performance Comparison across Uncertainty Quantification (UQ) Methods for Models Trained on CrackForest and DeepCrack Datasets	71
7. D435i Settings for Image Acquisition.	98
8. Model Performance Comparison for Segmentation Tasks	100
9. Performance of RGB-DNC Using the AF Fusion Model with Different Dropout Probabilities	100
10. Defect Measurement Demonstration from the ASU Pipe Dataset	102
11. Dataset Sizes	118
12. Encoder Layer Architectural Details with Channels Subdivision and Operation Parameters	120
13. CAM Block Details	121
14. Decoder Architecture	121
15. Effect of Incorporating Unlabeled Samples at Different Ratios to Labeled Samples in the NEU Test Set	123
16. Results for the NEU and MagTile Datasets Compared against Similar Models	125

Table	Page
17. Performance Comparison across Baseline Model Architectures	148
18. Impact of Regularization on F-1 Score	149
19. F1 Scores and Variances for Two Dropout-Based UQ Methods.....	150
20. Performance across Sentence Embedding Models	150

LIST OF FIGURES

Figure	Page
1. 1-D Constrained Regression Demonstrations with Value, Derivative and Bound Constraints	31
2. 1-D Constrained Regression Demonstrations Using PR-BNN with Value, Derivative and Bound Constraints	32
3. 1-D Constrained Regression Demonstrations with Variance Constraints outside the Training Data	32
4. Beam Configuration. The Observable Regions Are Given in the Dashed Boxes	34
5. Predicted Beam Deflection with BENN and BNN	34
6. Comparing the L1 Error on TPCF across Three Microstructure Generation Scenarios: Unconstrained, TPCF Constraint Only, and TPCF + Porosity Constraint	40
7. Microstructure Generation Using the Proposed Model after Training the Model with 260 Epochs with (Top Row) No Constraints (the Baseline CVAE Model) (Middle Row) TPCF Constraint (Bottom Row) TPCF + Porosity Constraints. Note that the Model Converges Faster with Two Constraints Compared to the Baseline or Just Using the TPCF Constraint	41
8. Overview of the Proposed Network Architecture Showing the Network Structure, Input, Predictions and Uncertainty. The Use of the Proposed Approach Allows the Model to Predict Tighter Crack Boundaries and Epistemic Uncertainties	53
9. Classification with Uncertainty in the Two Moons Dataset - Variation of Uncertainty and Entropy with Sample Size	57

Figure	Page
10. Classification with Uncertainty in the Two Moons Dataset - Variation of Uncertainty with Sample Size.	57
11. Demonstrative Examples of Detections in the CrackForest (Top-2 Rows) and the DeepCrack Dataset (Bottom-2 Rows). Images from Left to Right: Input, Prediction, Ground Truth, Epistemic Uncertainty, Aleatoric Uncertainty ...	65
12. Evaluation of Models Trained on the CFD Dataset across Training Samples on Both the CFD Test Set and the DeepCrack Test Set.	67
13. Evaluation of Models Trained on the CFD Dataset across Training Samples on the DeepCrack Test Set on the Cross-Entropy Trained Bayesian FCN Baseline and the Proposed B-BACN Model.	67
14. Uncertainty, F1 Score and Calibration Error on the DeepCrack and CFD Test Sets for Models Trained on the Two Respective Datasets across Dropout Ratios.	68
15. Effect of Varying the Number of Dropout Layers in the Network on Model Performance and Uncertainty.	69
16. Model Performance Comparison on CFD Dataset with Varying Training Samples: Epistemic and Aleatoric Uncertainty, F1-Score, and ECE.	70
17. Epistemic Uncertainty and F1 Score for a Model Trained on the DeepCrack Dataset: (Row-1) Evaluation on the DeepCrack Test Set with No Noise, (Row-2) Evaluation on the DeepCrack Test Set with Input Gaussian Noise $\sim U(0.0,1.0)$. (Row-3) Evaluation on the CrackForest Test Set with No Noise. (Row-4) Evaluation on the CrackForest Test Set with Input Noise \sim $U(0.0,1.0)$	73

Figure	Page
18. (A) Projection from the Image Plane to 3D World Coordinates Is Only Known Upto Scale (B) Triangulation Approach to Find the Corresponding Point on the Second Camera for a Point on the First Camera.	80
19. (A) Sensitivity of the Depth Map to Regions at Various Distances in (Mm) to Perturbations in Disparity, Focal Length, and Baseline. (B) Sensitivity of the Depth to Disparity Perturbations for Objects at Various Distances...	83
20. The Relationship between the Length of an Object in Image Coordinates and the World Coordinates Can Be Determined Using Similar Triangles....	83
21. RGB-DNC Data Pre-Processing Module: This Module Compresses the Channel-Wise Dimension of the DNC Input from 6 to 3 Channels.	88
22. Feature Descriptors for Various Rigid Objects Demonstrating the Various Invariances that Curvature Possesses - By Row from Top to Bottom: Flat Plate, Smooth Cylinder, Rotated Smooth Cylinder - Euler Angles $[15^\circ, 15^\circ, 0^\circ]$	91
23. Comparing the RGB-D and RGB-DNC Data Input Formats: RGB-DNC Provides a More Refined Representation of Defects at the Input Level.	92
24. Overview of the Proposed Network Architecture: Overall Network Architecture Diagram for Feature Fusion in Semantic Segmentation. Two Fusion Schemes Proposed in This Chapter Are Showcased Here.	93
25. Post-Processing of the Segmentation and Point-Cloud Maps to Extract Crack and Corrosion Defect Measurements Using Contour Analysis.	97
26. Demonstrative Examples of Detections in the ASU Pipe Dataset. Images from Left to Right: Input, Prediction, Ground Truth, Total Uncertainty ...	101

Figure	Page
27. Overall Methodology for Semi-Supervised Segmentation Using Interpolation Consistency in the Probability Space: Blocks with the Same Colors Denote Related Elements. Note that the Blue Box Denotes the Common Decoder Weights for Both the Labeled Data and the Interpolated Representation from the Unlabeled Pair. The Training Is to Be Done Concurrently for Both Labeled and Unlabeled Batches.	112
28. Cluster Assumption Demonstration by Computing the Patchwise Euclidean Distance between a Patch and Its Neighbors. Higher Values Are Indicated with Yellow: Left-Most Image Is the Input, Followed by the Distance Maps of the following: the Input Image, the Activation Map from Layer 4, and the Activation Map from the Final Layer of the Encoder.	113
29. Demonstrative Images from the Magnetic Tiles Defect Dataset	118
30. Demonstrative Images from the NEU Steel Surface Defects Dataset	118
31. NEU Test-Set Output Uncertainty as a Function of Input Gaussian Noise Level	123
32. Demonstrative Detections on the NEU Validation Set: (Left to Right) Image, Ground Truth, 5% Labeled, 10% Labeled, 20% Labeled, 50% Labeled, Fully Labeled.	124
33. Demonstrative Detections Illustrating the Effect of Adding Additional Unlabeled Data to the Set and Using the Consistency Loss. In This Case, the Comparison Is Made between the Fully-Supervised Case with 30 Labeled Samples, and the Semi-Supervised Case with 30 Labeled Samples and 570 Unlabeled Samples.	124

Figure	Page
34. Test-Set Performance on the NEU Test Set with and without Consistency Loss Ramp-Up	125
35. Model Uncertainty as a Function of Percentage of Labeled Data	126
36. Overall Network Architecture for Weather Threats Prediction	137
37. Distribution of Weather Parameters Used in the Generated Dataset	143
38. Representative Images from the GA-Situational Awareness Dataset	145

Chapter 1

INTRODUCTION

1.1 Overview

Engineering has witnessed a paradigm shift propelled by advancements in machine learning and data analytics. Among these, neural networks are beginning to emerge as powerful tools, offering significant potential for enhancing predictive accuracy and decision support (Pin *et al.*, 2020; Krizhevsky *et al.*, 2017). Recent advancements in convolutional neural networks have led to state-of-the-art results in various tasks such as visual recognition, driven by the growth in annotated data and hardware (Gu *et al.*, 2018). However, despite these advancements, several gaps remain in the realms of uncertainty quantification and the tailoring of these methods to real-world engineering problems. The escalating complexity and scale of data in engineering contexts underscore the need for models that not only predict with high accuracy but also quantify the uncertainty in these predictions (Abdar *et al.*, 2021). Uncertainty quantification is crucial in high-stakes scenarios such as structural health monitoring and decision support systems (Sankararaman and Mahadevan, 2009; Nasr *et al.*, 2018). In these contexts, the accuracy of predictions is not the only concern. Understanding the confidence and limitations of these predictions is equally important, as the consequences of errors can be significant. By quantifying uncertainty, engineers and decision-makers can better assess the risks associated with different actions or inaction, leading to more informed and safer choices. It allows for a more nuanced approach than simply relying on the most likely prediction by also considering the range of

possible outcomes and their associated probabilities. This is especially important in fields like structural health monitoring, where the integrity and safety of infrastructure are at stake, and in decision support systems that require downstream processing steps after the prediction before taking action. Machine learning works on the interplay between the data, the model, the optimization objective functions, and the evaluation metrics. The focus of this document is novel architectures, regularization methods, and data fusion techniques, which encompass the first three elements that will cause improvements in the fourth.

1.1.1 Data Handling

Deep learning in quality inspection systems often require a significant amount of annotated images, which can be costly and time-consuming to collect. This is particularly challenging for rare defects due to the scarcity of samples. One way to handle this is using simulated environments to generate training data (Gutierrez *et al.*, 2021). This helps reduce costs and puts forth directions for model prototyping and evaluations, provided the simulated data is close enough to what one might encounter in the real world.

Utilizing deep learning algorithms with vision systems has shown feasibility in automated industrial quality inspection (Rathod and Salehi, 2020; Cheng and Zhou, 2021; Yang *et al.*, 2020a; Liu and He, 2022). For instance, the integration of deep learning with vision systems for industrial quality inspection allows for efficient detection of product surface defects. The challenge for deep learning lies not merely in accessing voluminous data but in acquiring data of a sufficient quality and in understanding how to leverage this data effectively to enhance specific metrics critical

to task performance. The choice of data and how it is processed can profoundly influence the outcomes of machine learning models. In traditional machine learning approaches, feature engineering played a pivotal role. It involves extracting meaningful information from raw data to improve model performance, particularly in specialized tasks where domain knowledge can guide the selection of relevant features. However, this approach has limitations, especially in handling intra-class variability, where the characteristics within a single class vary significantly. This variability can cause models reliant on manual feature engineering to underperform or require constant adjustment to maintain performance. Deep learning, in contrast, simplifies this process by employing layers of parameters that automatically learn to transform data into relevant features. This marks a departure from the manual, often labor-intensive process of feature engineering, enabling models to scale and adapt to a broader range of tasks with diverse data sets.

Benchmark datasets have been created to aid in training machine learning models for various industrial tasks. These benchmarks are diverse in size, ranging from small (Shi *et al.*, 2016) to massive. These large datasets are often found in applications such as autonomous driving and generic object segmentation (Silberman and Fergus, 2011; Cordts *et al.*, 2016). In line with these efforts, the work done in this dissertation has resulted in the creation of two benchmark datasets specifically designed for semantic segmentation and classification tasks.

1.1.2 Data Fusion

The combination of multiple data modalities has seen the development of a lot of foundational ideas in the controls and statistics literature. Methods like Kalman

filtering and fuzzy logic have been used in many applications to combine information from multiple sensors. (Alofi *et al.*, 2017) provides a comprehensive comparison of various data fusion techniques, including the Kalman filter and Bayesian Methods. Alongside the adaptation of deep learning, a lot of these ideas are seen to transition into the deep learning research literature (Stahlschmidt *et al.*, 2022; Gao *et al.*, 2020).

Industrial informatics applications can take advantage of a variety of sensor technologies to enhance their capabilities. For instance, stereovision for part quality inspection was studied in (Malassiotis and Strintzis, 2003). Another example is from (Zhao *et al.*, 2021), where they present a method for general object recognition using multi-source data fusion. It integrates RGB, and Depth image features through a feature fusion module, employing residual learning to reduce model parameter complexity. A common thread to this literature is the variety of schemes employed to induce improved empirical performance on benchmark datasets. With this in mind, this dissertation focuses on RGB-Depth data fusion for the detection of defects in gas pipelines. Specifically, the focus is on the detection and quantification of both cracking and pitting corrosion defects that occur in pipelines. What makes this problem important is the interacting nature of these threats in the gas pipeline. The existence of multiple defects near one another causes the failure probability to increase significantly more than if only one defect was present individually. This is exacerbated by the presence of pitting defects that can occur in various configurations and densities. Current pipe inspection techniques rely on ultrasonic sensors (Iyer *et al.*, 2012; Sun *et al.*, 2023) and magnetic flux leakage detectors (Shi *et al.*, 2015), which require human analysis and face limitations like sensitivity to specific defect types and operational constraints. Industry trends are moving towards vision-based methods, which can detect textural and geometric changes more intuitively, potentially

leading to more cost-effective and versatile robotic solutions that can enhance existing inspection protocols.

1.1.3 Semi-supervised Learning in Industrial Informatics

Many industrial applications have the potential to utilize massive training datasets for machine learning models. While this data has definitely been utilized well, it has often been done at the cost of data labeling and feature engineering. This increases both the cost and time to productionize machine learning workflows. Models that require the use of fully labeled (x,y) pairs of training samples are called supervised learners. The other end of this spectrum, unsupervised learning, does not require any labels at all to learn a function to map x to y . To take advantage of supervisory signals, while not having to label the entire dataset manually before training is the motivation behind semi-supervised learning algorithms. Semi-supervised learning algorithms rely on the ability to extract patterns and insights from the unlabeled data conditioned on the strong signal from the labeled data, using them to enhance the learning process. By capitalizing on the inherent structure of this unlabeled data, these models can improve their accuracy and robustness, surpassing the performance of models trained solely on limited labeled datasets. Semi-supervised models can even adapt to an influx of data that is not static, but evolves, continuously refining their predictions and providing up-to-date predictions (Din *et al.*, 2020). There are a multitude of semi-supervised learning methods specific to various application areas, but the scope of this dissertation is with respect to the utilization of semi-supervised models for manufacturing inspection tasks with imaging data.

1.1.4 Uncertainty Quantification

Traditional neural network models, while effective in various applications, often fall short in providing a comprehensive framework for uncertainty quantification. This limitation becomes more pronounced in industrial informatics, where the data is not only vast but also often imbalanced, noisy, and incomplete. In such scenarios, uncertainty quantification for the predictions become crucial. Addressing this gap, this dissertation utilizes Bayesian inference techniques that imbue neural networks with the capacity to quantify and manage uncertainty in a principled manner. The integration of Bayesian principles with neural network architectures enables the development of models that are not only predictive but also provide a measure of confidence in their predictions, a feature critical for decision-making in engineering applications. Moreover, this work proposes novel neural network architectures and regularization techniques specifically designed for the challenges encountered in industrial informatics problems such as defect detection. These methods are empirically tested and validated using benchmark datasets and custom-built, in-house datasets, ranging from the detection of structural defects to the provision of decision support for general aviation pilots.

1.2 Outline of the Dissertation

The motivation behind this dissertation stems from the need to develop, analyze and refine Bayesian neural network models that are not only theoretically sound but also practically applicable in addressing pressing challenges in engineering today. By focusing on uncertainty quantification and data fusion, this research seeks to bridge

the gap between advanced statistical methods and real-world engineering applications, ensuring that decisions made on the basis of these models are both reliable and robust.

This dissertation aims to develop uncertainty-aware neural network frameworks that enhance the capabilities of engineers in detecting, analyzing, and making informed decisions under conditions of uncertainty. The following chapters delve into various aspects of this overarching goal, each addressing specific challenges and proposing solutions that advance the understanding and application of neural networks in engineering problems.

Chapter 2 provides background on some of the foundational concepts used throughout this dissertation.

Chapter 3 proposes a framework that can integrate different types of constraints on neural network predictions in a principled, Bayesian manner. The need for deep learning models to incorporate programmable constraints in their outputs is motivated by their use in surrogate models, learning with limited data and partial information, and in similar situations where the model behavior needs to be flexible. This framework is based on Maximum Entropy (MaxEnt) for constraining Bayesian Neural Network predictions, demonstrating an ability to constrain values, derivatives, and variance of predictions.

In Chapter 4, an analysis of popular Bayesian Deep Learning techniques in the context of crack image segmentation is presented. Specifically, the detection and monitoring of structural defects using uncertainty-aware neural networks, such as those occurring in buildings and roads, is presented. Traditional methods often struggle with the challenges associated with accurately detecting defect morphology and dealing with intra-class variability. This gap in capability underscores the necessity for innovative

approaches that not only embrace the complexity of these tasks but also provide robust, uncertainty-aware solutions.

Furthermore, the integration of such advanced techniques in practical scenarios, like in-line inspection of gas pipelines, presents its own set of unique challenges. These include the need for models that can effectively fuse diverse data types, handle limited labeled data, and provide interpretable, actionable insights in dynamic, often critical environments. To this end, the following chapters focus on these specific areas.

Chapter 5 proposes an end-to-end semantic segmentation model for gas pipeline defect detection that both leverages the geometric considerations of the scene and also uses learned features vis-a-vis deep learning, with a combination of optimization objectives.

Chapter 6 proposes a new semi-supervised method to deal with limited labeled data in the context of manufacturing defect detection. The method that can take a small labeled set of data, and a larger set of unlabeled data, to produce fine-grained defect segmentation that comes close to the performance of fully supervised methods.

Finally, this dissertation also focused on building a framework for improving situational awareness in aviation in Chapter 7. While seemingly distinct in its aviation focus, this work applies the principles seen in the preceding chapters. It extends the concepts of Bayesian uncertainty quantification and neural network modeling into the realm of aviation, specifically tailored for General Aviation (GA) pilots. This chapter showcases the practical applicability and versatility of the neural models in a high-stakes environment, along with highlighting the potential of commercially available image sensors to reduce the cost of acquiring data compared to more expensive alternatives. The decision support system designed for GA pilots leverages data fusion and handling variable data quality and quantity. The application provides

further evidence of the flexibility and broad applicability of uncertainty-aware deep learning frameworks and uncertainty quantification methods developed throughout this dissertation.

Chapter 2

BACKGROUND

This chapter outlines some technical concepts foundational to this dissertation. Initially, it provides an overview of neural networks and explores the rationale for using Bayesian Neural Networks. Subsequently, it discusses several methods for performing Bayesian inference in neural network models.

2.1 Neural Network Models

Regression models assume a linear relationship between input variables (features) and a target variables, formulated as:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n + \epsilon(x_1, \dots, x_n)$$

where y is the target, x_1, x_2, \dots, x_n are inputs, $\beta_0, \beta_1, \dots, \beta_n$ are coefficients, and ϵ represents a noise function that can either be independent or have input dependencies as shown in the equation.

Neural networks can be thought of as layers of regression models that build upon each other, introducing the ability to approximate more complex functions:

$$z^{[1]} = W^{[1]}x + b^{[1]}, \quad a^{[1]} = \sigma(z^{[1]}) \quad (2.1a)$$

$$z^{[2]} = W^{[2]}a^{[1]} + b^{[2]}, \quad a^{[2]} = \sigma(z^{[2]}) \quad (2.1b)$$

\vdots

$$z^{[L]} = W^{[L]}a^{[L-1]} + b^{[L]}, \quad \hat{y} = \sigma(z^{[L]}) \quad (2.1c)$$

where $W^{[l]}$ and $b^{[l]}$ are the weights and biases of layer l , σ is a non-linear mapping from the output of the hidden layers, and \hat{y} represents the output prediction. These mappings are called activation functions.

2.1.1 Inductive Biases in Neural Networks

Going back to the regression example, it is easy to see that the model induces strong assumptions on the set of functions that can be admitted as solutions to the fitting problem given some data. Linear regression forces the output to be linearly related to the inputs, with an additive noise component that may vary with each input. Neural network models can incorporate many interesting inductive biases, a few of which that are most relevant to this dissertation will be discussed in this section.

The literature sees commonly used functions such as sigmoid, $\sigma(x) = \frac{1}{1+e^{-x}}$ or $\text{ReLU}(x) = \max(0, x)$ as non-linear activation functions. These activation functions can serve to modulate the functional behavior of the neural network, introducing what are known as inductive biases into the model. For instance, the sigmoid squashes the values of the outputs between 0 and 1, which makes it a useful probabilistic tool. The ReLU is a non-linearity that is commonly used in the latent (hidden) layers of the neural networks.

Another source of incorporating inductive biases into the model can come from the organization of the parameters. While the basic neural network in Equation 2.1 has each element of the output multiplied by different weights, specific elements in the output can also have a common weight associated with it. This type of inductive bias motivates the use of the popular convolutional neural network (CNN) (LeCun *et al.*, 1995) architecture. CNNs exhibit a significant inductive bias specifically crafted

to handle grid-like data, such as images, through several key architectural features including local receptive fields (Luo *et al.*, 2016) and spatial hierarchies (Kavukcuoglu *et al.*, 2010; Zhou *et al.*, 2016). Local receptive fields refer to the small, localized areas of the input data, such as a patch of pixels in an image, that each neuron in a convolutional layer observes. This structure enables the neuron to detect patterns such as edges or textures within this confined space, contributing to the network’s ability to recognize local features without the influence of the entire image. Such localized processing simulates how human vision works, focusing on specific segments before integrating the information for broader understanding. Spatial hierarchies organize these local features into increasingly complex representations at each subsequent layer of the network. The first layer may identify simple patterns like edges, the next layer assembles these into contours, and higher layers may interpret these contours as parts of objects. This hierarchical processing facilitates the abstraction of high-level features from raw pixel data, allowing CNNs to build an intricate understanding from simple to complex elements within an image. The shared weights across these hierarchical layers promote translational invariance, meaning once a feature is learned at one position, it can be recognized anywhere in the image.

Neural network training involves optimizing weights and biases using backpropagation Rumelhart *et al.* (1986) and algorithms like stochastic gradient descent Bottou *et al.* (1991) to minimize the loss between predicted and actual outputs.

Standard neural networks, however, do not account for uncertainty in their predictions, providing so-called point estimates obtained through the maximum likelihood estimation process. This limitation is addressed by Bayesian Neural Networks, which incorporate Bayesian probability principles to manage uncertainty and provide more reliable predictions, discussed in Section 2.3.

2.2 Uncertainty Quantification

Prior to discussing the specifics of Bayesian Inference in Neural Networks, a discussion that motivates why you need Bayesian reasoning in the context of industrial risk assessment and decision support is important.

For instance, decision support systems play a crucial role in identifying and mitigating risks associated with defects. These systems rely heavily on accurate defect detection mechanisms to prevent potential failures and ensure product quality. Risk assessment in this scenario involves evaluating the likelihood and potential impact of defects, which can vary greatly depending on the industry sector, from minor cosmetic issues in consumer products to critical structural failures in aerospace components. The effectiveness of these assessments is directly tied to the quality of the data and the precision of the prediction models used. By incorporating uncertainty quantification into neural networks, decision support systems can better assess risks by not only predicting the presence of defects but also providing a measure of confidence in these predictions. This dual capability is essential for making informed decisions about whether to reject a product, rework it, or accept it as is, thereby optimizing resource allocation and minimizing potential losses.

To effectively implement uncertainty quantification in neural networks, it is crucial to distinguish between the different types of uncertainty. Broadly, uncertainties can be categorized into two types: aleatoric and epistemic.

Aleatoric uncertainty arises from inherent randomness in the system or environment. This type of uncertainty is irreducible and can be attributed to factors like sensor noise or inherent variability in material properties. In the context of decision support

systems, accounting for aleatoric uncertainty ensures robustness against natural variations in the input data.

Epistemic uncertainty originates from inadequate knowledge or insufficient data. This type of uncertainty can be mitigated by acquiring more data or refining the model architecture. In industrial contexts, the significance of epistemic uncertainty is profound as it mirrors the confidence in the model’s predictions, which in turn affects risk management and decision-making.

Within epistemic uncertainty, it is valuable to distinguish between parameter uncertainty and model uncertainty, as each influences the reliability of model predictions differently:

Parameter uncertainty pertains to the uncertainty associated with the parameters of a specific model. Variations in training data or conditions can lead to different parameter estimations, thereby influencing the predictions made by the model. Such uncertainty can typically be quantified through Bayesian inference, where model parameters are considered as random variables defined by probability distributions.

Model uncertainty emerges from the selection of the model structure and its associated inductive biases. Various model architectures fit the data differently, potentially overlooking or misrepresenting certain correlations. The bias-variance tradeoff offers insights into this: For instance, a neural network that is under-parameterized might not capture all necessary features, while an overly complex model might model noise as if it were signal. Tackling model selection bias generally involves using cross-validation, applying model comparison criteria, or deploying ensemble techniques that combine the outputs of several models to improve generalization and reduce the likelihood of an inappropriate model choice. In this dissertation, particular focus is given to Bayesian Neural Network models for parameter and aleatoric uncertainties.

2.3 Bayesian Neural Networks

2.3.1 Bayesian Parameterization of Neural Networks

Bayesian Neural Networks arose out of a need to have a principled alternative to point-estimates of parameters in standard neural networks. Given a neural network $f(\theta)$, and dataset $D = (x_i, y_i)_{i=1}^n$, one can formulate the process as updating a prior belief $p(\theta)$ using a likelihood $p(D|\theta)$, to obtain a posterior $p(\theta|D)$. Note that the prior distribution of the parameters are only part of the overall prior hypothesis. The inductive biases of the model, as discussed in the previous section are an implicit prior imposed on the model itself. The prior updating scheme is given by Bayes rule:

$$p(\theta|D) = \frac{p(D|\theta) \cdot p(\theta)}{p(D)} \quad (2.2)$$

In the above equation, the denominator $p(D)$ is the marginal likelihood of the data, that is obtained after marginalization of the parameter θ . While the Bayes rule is valid for any family of models, implementing updating schemes for neural networks presents a challenge. The dimension of the parameter set θ can be large, leading to computational difficulties in calculating the marginal likelihood. Moreover, the goal of inferring parameters is to predict on test samples x^* . However, one again confronts a need to marginalize the parameters using $p(y^*|x^*, D) = \int p(y^*|x^*, \theta)p(\theta|D)d\theta$ to get an averaged prediction. Sampling approaches such as Markov Chain Monte Carlo (MCMC) are often used in smaller models to get accurate estimates of the true posterior distributions. However, for neural networks, this sampling approach is disfavored due to excessive computational requirements. Instead, variational inference techniques are used.

VI approximates the posterior by finding a proposal distribution $q_\phi(\theta)$ that min-

minimizes the KL-divergence between the two. The KL divergence equation is given by:

$$\text{KL}(q_\phi(\theta)||p(\theta|D)) = E_q[\log(q_\phi(\theta))] - E_q[\log(p(\theta|D))] \quad (2.3)$$

From Bayes rule, it is seen that the second term is still dependent on the marginal likelihood, which is intractable. However, $p(D)$ is independent of the proposal distribution $q_\phi(\theta)$. Taking advantage of this fact, variational methods maximize an equation that is proportional to the KL-divergence, the Evidence Lower Bound (ELBO):

$$\begin{aligned} \text{ELBO}(q) &= E_q[\log(p(D|\theta))] + E_q[\log(p(\theta))] - E[\log(q_\phi(\theta))] \\ &= E_q[\log(p(D|\theta))] - \text{KL}(q_\phi(\theta)||p(\theta)) \end{aligned} \quad (2.4)$$

The two components present in the final form consist of an expected log likelihood term that is data-dependent, and a term that encourages the proposal distribution to not deviate from the prior.

The ELBO equation has gradients that can be estimated using Monte Carlo sampling of the parameter θ^i from the approximated posterior after each iteration of the training loop.

$$L_{VI} = \sum_{i=1}^n \log(q_\phi(\theta^i)) - P(\theta^i) - P(D|\theta^i) \quad (2.5)$$

It is important to recall here that the parameter θ is a random variable. Directly performing backpropagation using realizations of θ implies that one needs to take the gradient of the expectation of the realizations. In many cases, the gradient of the expectation is the expectation of the gradient. But when the sampling distribution itself is parameterized by another random variable, the gradients are estimated using a reparameterization trick (Blundell *et al.*, 2015):

$$\theta = \mu + \log(1 + \exp(\rho))\epsilon, \epsilon \sim N(0, I) \quad (2.6)$$

The above reparameterization allows for computing the overall loss function given in Equation 2.5 by calculating the gradient with respect to the mean and the variance using backpropagation, and update these parameters with commonly used optimization methods such as Stochastic Gradient Descent (SGD).

2.3.2 Dropout Variational Inference

Another commonly used VI approach is the MC-Dropout (MCD). MC-Dropout performs VI with a specific choice of the approximating distribution $q_\theta(w|D)$. In this case, q_θ is a distribution over the weight matrices whose columns are randomly set to zero. This is commonly referred to as dropout in the deep learning literature:

$$\Theta_i = M_i \cdot \text{diag}([z_{i,j}]_{j=1}^{K_i}) \quad (2.7)$$

$$z_{i,j} \sim \text{Bernoulli}(p_i), i = 1, \dots, L, j = 1, \dots, K_{i-1}$$

In the above equation, M_i is the deterministic weight matrix for a layer L_i , and Θ_i is the stochastic parameter set induced by the sampling of an iid Bernoulli random vector z_i . The MC-Dropout inference procedure minimizes the KL-Divergence between $q_\phi(\theta|D)$ and a . Details on this derivation are given in the Appendix of Gal and Ghahramani (2016). After inference, the uncertainty is evaluated by averaging N forward passes during prediction. This is equivalent to drawing N samples from the set of parameters defined in the model posterior and evaluating a function using each of those samples. The sample mean of the predictive distribution approaches the population means at large sample sizes, with a spread that is dictated by the form of the function resulting from each dropout sampling iteration.

The dropout probability p_i in MC-Dropout is a hyperparameter that requires expensive grid search to obtain a good epistemic uncertainty. The objective function

proposed by (Gal *et al.*, 2017) optimizes the dropout probability p using an entropy term that penalizes high dropout probabilities. This objective is derived analytically using a discrete quantised gaussian prior, which results in the following KL divergence objective:

$$KL(q_\phi(\theta)||p(\theta))\alpha\frac{l^2(1-p)}{2}||M||^2 - KH(p) \tag{2.8a}$$

$$H(p) = -p\log p - (1-p)\log(1-p) \tag{2.8b}$$

The inclusion of the entropy term $H(p)$ reflects the information-theoretic nature of uncertainty. By discouraging extreme values of p , the model is less likely to become either too confident (overfitting) or too uncertain (underfitting) about its predictions. The coefficient α acts as a scaling factor that adjusts the strength of the regularization, effectively controlling the trade-off between the model complexity and the entropy term. The l^2 regularization term $||M||^2$ penalizes large weights, promoting generalization by preventing over-reliance on any single feature.

BAYESIAN ENTROPY NEURAL NETWORKS FOR PHYSICS-AWARE PREDICTION

3.1 Introduction

Neural networks have achieved remarkable success in various domains, ranging from image recognition to natural language processing (Goldberg, 2016; Khan *et al.*, 2020). Despite these advancements, traditional deep learning approaches often fall short in scenarios that demand adherence to specific constraints (Zhu *et al.*, 2019; Hosseini-Asl *et al.*, 2015) or a deeper understanding of model uncertainty (Wilson and Izmailov, 2020). This limitation becomes particularly pronounced in fields such as surrogate modeling, learning with limited or partial data, and applications where flexible adaptation to non-sample based information is crucial. For surrogate models, the neural network model is expected to conform to behavior governed by the system model, and without enough data across the domain of interest, neural network performance falls short.

Raissi *et al.* (2019) presents physics-informed neural networks (PINNs) that directly incorporate physical laws in the form of differential equations, directly into the loss function of the neural network. This is achieved by constructing a composite loss function that not only measures the difference between predictions and data but also quantifies the deviation from the specified physical laws. PINNs are specialized models that have been shown to model differential equation constraints accurately using regularization.

Wang *et al.* (2020b) presents a method for information fusion combining the maximum entropy (ME) method with the classical Bayesian network, termed as the Bayesian-Entropy Network (BEN). This method is particularly adept at handling various types of information for classification and updating, such as point data, statistical information, and range data. The BEN method extends the Bayesian approach by integrating additional information in the form of constraints into the entropy part, while the Bayesian part handles classical point observation data. This integration allows for a more comprehensive approach to modeling and decision-making. The BEN method demonstrates a flexible adaptation to non-sample based information, making it particularly relevant in fields like surrogate modeling and learning with limited or partial data. For example, Wang *et al.* (2021d) applies this method to the Gaussian Process model by adding information as constraints. In this chapter, this is extended to apply to neural network models, showcasing its potential in tasks such as information constraints for high dimensional problems such as microstructure generation, but also in controlling the predicted uncertainty, which is an important avenue of research. The reason for the latter is motivated by wanting to output high uncertainty for predictions outside the training domain, or if the information constraints are conflicting, as will be shown in the later stages of this chapter.

Huang *et al.* (2022) and Yang *et al.* (2020c) are two contemporary papers that are closest to this work. Huang *et al.* (2022) focuses on embedding soft and hard knowledge constraints into the posterior, offering enhanced model robustness and adaptability. Yang *et al.* (2020a) modifies the prior, where the prior is optimized to comply with the constraints. Where OC-BNNs (Yang *et al.*, 2020c) primarily focus on defining equality and inequality constraints by multiplying likelihoods and priors in the probability space and PR-BNNs directly regularizes the posterior predictive distribution to comply

with the constraint, in this chapter, a framework that generalizes these approaches by integrating a broad range of constraints using the Maximum Entropy (MaxEnt) principle is proposed.

The essence of Bayesian Entropy Neural Networks (BENN) lies in its ability to integrate the Maximum Entropy (MaxEnt) principle with the Bayesian framework, resulting in a model that not only provides constrained and uncertain predictions but also remains computationally feasible. The methodological underpinnings of Bayesian Neural Networks (BNNs), founded on principles of Bayesian inference, present an elegant solution to the over-reliance on point estimates inherent in traditional neural networks. By leveraging posterior distributions over parameters, BNNs offer a more nuanced view of model uncertainty, essential in many real-world applications. However, the practical implementation of Bayesian updating in neural networks is challenged by computational limitations, particularly when dealing with large parameter spaces. To address this, Variational Inference (VI), an approach that significantly reduces computational overhead by approximating posterior distributions, is utilized. This approximation is crucial for enabling the application of BNNs in more complex and realistic scenarios. Works such as Wang *et al.* (2021a) use a sequential optimization approach to iteratively find model parameters for the Gaussian Process and the Lagrange multipliers. However, neural networks provide us with backpropagation machinery to handle simultaneous optimization under constraints. Therefore, this work utilizes the Modified Differential Method of Multipliers (MDMM) to efficiently learn constraint weights.

The rest of the chapter is organized as follows:

1. Section 3.2 provides background information on the Maximum Entropy method,

and ends with a description of the Bayesian Entropy Neural Network (BENN) framework.

2. Section 3.3 covers three experiments:
 - a) 1D Regression: Implementing value, derivative and predictive variance constraints.
 - b) Beam deflection: The 1D regression section is supplemented by demonstrating the performance of the BENN approach in a beam deflection problem, where part of the domain is observed.
 - c) Microstructure generation: This experiment showcases the benefit of constrained learning to enhance the performance of microstructure generation using suitable constraints on the microstructure properties.
3. Section 3.4 summarizes the work done in this chapter and the importance of the main results obtained.

3.2 Method

3.2.1 Bayesian Entropy Method for Incorporating Constraint Information

The method of maximum entropy, seen in (Giffin *et al.*, 2007) was proposed by Jaynes to perform inference of model parameters subject to constraints. In other words, MaxEnt picks parameters that maximize the entropy of its distribution insofar as the constraints allow. As a corollary, it is also obtained that given no information about the parameters, a flat distribution is the one that maximizes entropy. This line of reasoning suggests that this method fits well within the Bayesian updating paradigm, as constraints are indeed a form of information about the parameters.

If we have a joint prior $p(\theta), x$ on a model given some information F about the model parameters, we can maximize the entropy between the prior and the posterior update considering the constraints as an optimization problem:

$$\begin{aligned} \operatorname{argmin}_{\theta} - \int P(\theta, x_c, x) \log \frac{P(\theta, x_c, x)}{P(\theta, x)} dx d\theta \\ \text{s.t. } \int dx d\theta p(\theta, x_c, x) = 1 \\ \int p(x_c, x, \theta) f(\theta) d\theta dx = F \end{aligned} \quad (3.1)$$

The above equation formalizes the entropy maximization problem subject to a constraint on the value of p and a normalization constraint on p . The solution is derived using the method of Lagrangian multipliers. If one extends the above constrained system to m constraints, the objective is to solve for $m + 1$ multipliers:

$$L(p, \lambda, \nu) = \int p_i \log(p_i) dp - \sum_{k=1}^m (\lambda_k \int (p_i f_k(x_i) - F_k) dp - \nu \int (p_i - 1) dp) \quad (3.2)$$

The solution takes the form of an exponential function, normalized by a partition function Z :

$$\begin{aligned} p(\theta|x, x_c) &= \frac{1}{Z(\lambda_1, \dots, \lambda_m)} \cdot \exp\left(-\sum_{k=1}^m \lambda_k f_k(x)\right) \\ Z(\lambda_1, \dots, \lambda_m) &= \int \exp\left(-\sum_{k=1}^m \lambda_k f_k(x)\right) \end{aligned} \quad (3.3)$$

From the above description, it is seen that the inference procedure requires a prior that captures no new information apart from the constraints and the data. Therefore, one choice to model this is to use a uniform prior. For a distribution $q_\phi(\theta)$ and a uniform prior distribution $p(\theta)$, the Kullback-Leibler (KL) divergence from $q_\phi(\theta)$ to $p(\theta)$ can be calculated as:

$$D_{KL}(q_\phi(\theta)||p(\theta)) = \int q_\phi(\theta) \log \frac{q_\phi(\theta)}{p(\theta)} dw \quad (3.4)$$

With $p(\theta) = \frac{1}{|\Theta|}$ for all $\theta \in \theta$, the KL divergence simplifies to:

$$D_{KL}(q_\phi(\theta)||p(\theta)) = \int q_\phi(\theta) \log(q_\phi(\theta) \cdot |\Theta|) d\theta \quad (3.5)$$

This can be decomposed into:

$$D_{KL}(q_\phi(\theta)||p(\theta)) = \int q_\phi(\theta) \log q_\phi(\theta) d\theta + \log |\Theta| \int q_\phi(\theta) d\theta \quad (3.6)$$

Given that $q_\phi(\theta)$ is a probability distribution, the second integral evaluates to 1, yielding:

$$D_{KL}(q_\phi(\theta)||p(\theta)) = \int q_\phi(\theta) \log q_\phi(\theta) d\theta + \log |\Theta| \quad (3.7)$$

The integral represents the negative of the entropy of q_ϕ , denoted as $-H(q_\phi)$. Therefore, the KL divergence can be rewritten as:

$$D_{KL}(q_\phi(\theta)||p(\theta)) = -H(q_\phi) + \log |\Theta| \quad (3.8)$$

Since $\log |\Theta|$ is constant, the negative KL divergence is effectively proportional to the entropy of the variational distribution. Such a measure has indeed been concurrently proposed by de Mathelin *et al.* (2023), where the model uses just the entropy of the proposal distribution for KL-divergence minimization. This is motivated by the above reasoning using the MaxEnt principle. Effectively, MaxEnt ensures that the prior assumptions are minimal, as the uniform prior contributes only a constant term to the KL divergence, making no further assumptions about the distribution of the parameters.

3.2.2 Bayesian Entropy Neural Networks

Building on the MaxEnt principle and Bayesian Neural Networks (BNNs), this section introduces Bayesian Entropy Neural Networks (BENN). The previous section

modeled constraints on the posterior distribution. Now, constraint enforcement on both the posterior predictive and the parameters is demonstrated using this framework. Given a Bayesian Neural Network $f_{BNN}(\theta)$, prediction \hat{y} and a constraint function g , the learning objective can be formulated as:

$$L_{VI} = \operatorname{argmin}_{\theta}(KL(q_{\phi}(\theta)||p(\theta)) - E_q(\log p(D|\theta)) + \lambda \cdot g(x)) \quad (3.9)$$

From Section 3.2.1, it is seen that the constrained optimization problem leads to an updated posterior of the form given by Equation 3.3.

BENN gives us an easy way to sample from the posterior distribution, so one can extract N samples from the posterior to evaluate the constraint. This is similar to the sampling approach proposed by Wang *et al.* (2021a).

A loss that includes the constraint as a regularizer may not always result in a local minimum, but can also be a stationary point. To improve this behavior, a more efficient updating scheme is used for parameters and the Lagrangian multipliers. This is done using a Modified Differential Method of Multipliers approach (MDMM) Platt *et al.* (1987) approach, that is found to be effective for regression. MDMM adds an extra damping term similar to penalty methods to the loss function, and has the following form:

$$\operatorname{argmin}_{\theta}L(p, \lambda, c) = \operatorname{argmin}_{\theta}(L_{VI} + \lambda g(x) + \frac{c}{2}g(x)^2) \quad (3.10)$$

Compared to (Wang *et al.*, 2021a) work on Bayesian Entropy Gaussian Process (BEGP), the BENN leverages the high dimensional representation capacity of neural networks, and uses the same underlying principle to enforce constraints. Moreover, the optimization framework obtained as a result of the backpropagation algorithm makes it possible to simultaneously optimize for both the model parameters and the

Lagrangian multipliers. Gaussian Processes, however, have inherent advantages in uncertainty quantification that Bayesian Neural Networks do not, and this chapter attempts to incorporate the principles from MaxEnt to demonstrate constraining predictive variance as well.

3.3 Experiments

3.3.1 1D-Regression

In this section, the data, architecture and loss used for the regression problem is first described. Following this, a demonstration of value and derivative constraints is shown. Finally, demonstrations to constrain the behavior of the predictive variance for 1-D regression are shown.

The dataset for the 1D-regression problem models a polynomial function with added noise and is defined over two distinct regions. In both regions, the function is given by:

$$y = p_0x^2 + p_1x + p_2 + 0.15 \sin(2\pi x) + \mathcal{N}(0, \sigma) \quad (3.11)$$

where the coefficients are $p_0 = 0.2$, $p_1 = 0.05$, and $p_2 = 0.01$. The first region, from $x = -3$ to $x = -2$, has a noise level with standard deviation $\sigma = 0.01$. The second region, from $x = 2.0$ to $x = 3.0$, has a higher noise level with standard deviation $\sigma = 0.1$. The training data X_{train} and corresponding labels Y_{train} consist of samples from both regions.

The architecture of the neural network used in these experiments is a Bayesian Neural Network (BNN). The network comprises one hidden layer with 100 units and employs the Rectified Linear Unit (ReLU) as the activation function. The output

layer of the network is designed to produce two outputs: the mean prediction of the target variable and a parameter termed as $\log(\text{noise})$ or σ , which represents the log-transformed noise level of the predictions. For the expected likelihood loss, it is assumed that the data generated is from a Gaussian, and a negative log likelihood loss is minimized:

$$\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}, \sigma) = \frac{1}{N} \sum_{i=1}^N \left(\frac{(y_i - \hat{y}_i)^2}{2 \exp(\sigma)} + \frac{\sigma}{2} \right) \quad (3.12)$$

Here, \mathbf{y} represents the actual values, $\hat{\mathbf{y}}$ denotes the predicted mean values, N is the number of data points, and σ is the log-noise parameter. This loss function not only penalizes the deviation of predictions from the actual values but also incorporates the uncertainty associated with the predictions as indicated by σ .

From the point of view of Maximum Entropy (MaxEnt), one seeks to make minimal assumptions about the nature of the model before training. In the Bayesian framework, this can be achieved by using a prior that is uniform over the space of parameters.

3.3.1.1 Incorporating Value and Derivative Constraints to the Prediction

In this section, the effects of incorporating additional constraints, specifically value and derivative constraints, into the 1D regression model is demonstrated. These constraints are particularly focused on areas outside the training data range. Several experiments to understand how these constraints influence the model's predictions are conducted, especially in regions where training data is sparse or absent.

3.3.1.1.1 Value Constraints Outside Training Range.

In the first demonstration, the two value constraints are placed at $x = 5.0$ and $x = 7.5$, both outside the training data range. This setup allows us to observe the model’s response to multiple external constraints and their impact on the overall prediction accuracy. The results, shown in Figure 1a, shows that the proposed approach is able to accurately capture the value constraint. The compliance of the network to the constraints is strongly satisfied compared to the data, owing to the representational capacity of the model.

3.3.1.1.2 Variance constraints

The BE technique can encode any type of information as a constraint. This includes constraints on the predictive variance. While this is generally modulated by the data using the Gaussian likelihood term, it is possible to enforce constraints on its behavior given expert knowledge about it. Examples for such cases might be to specify the level of decay in prediction confidence outside the training data range. There are two options for implementing this using the BE method: The weights, and the output. Figure 3 showcases three examples of arbitrary variance constraints on the output variance neuron, applied in regions outside the training data, with increasing uncertainty.

3.3.1.1.3 Conflicting Constraint and Variance Constraint at Same Location.

Model behavior in the presence of conflicting constraints is shown in Figure 1b. Two different value constraints are applied at the same location $x = 7.5$. This experiment is designed to evaluate how the model handles conflicting information and to understand the role of variance constraints in such situations. The variance at the constraint location is meant to express the uncertainty in the data provided in the form of a constraint. It is also reasonable to want to express arbitrary values of uncertainty at such points. However, the model needs to account for the variance that is admissible in these situations, so the predictive variance is constrained to be proportionate to the uncertain value constraints.

3.3.1.1.4 Bound Constraints

Next, an evaluation of the model with bound constraints to ensure that the predictions fall within a certain range is shown. For a given input x , the bound constraints are defined by two functions, representing the upper bound $\text{ub}(x)$ and lower bound $\text{lb}(x)$. These are given by:

$$\text{ub}(x) = 1.0, \quad \text{lb}(x) = 0.5, \quad \forall x \in [-0.5, 0.5] \quad (3.13)$$

where x is the input to the model. n bound constraints are enforced, each represented as a hard constraint. For each test input $x_i \in X_{\text{test}}$, the constraints are formulated as:

$$\text{lb}(x_i) \leq \text{pred}_i \leq \text{ub}(x_i), \quad \forall i \in \{1, \dots, n\}. \quad (3.14)$$

Like the equality constraint, the infeasibility for the bounds is quantified by how far a prediction is from satisfying the constraints. The bound constraints are assessed

by computing the deviation of the bounds from the original prediction. The variance constraints modify the predicted aleatoric variance to minimize the absolute difference between the predicted and expected variance values.

Table 1 showcases a comparison of evaluation scores produced by PR-BNN (Huang *et al.*, 2022) and BENN on constrained regression experiments. The evaluation scores are computed as the absolute value of the constraint violations, $|y_{pred} - y_{gt}|$, where y_{pred} is averaged from 250 evaluations of the model. In the table, Inf-1, Inf-2 indicate the constraint violations of each constraint. For instance, the value constraints for both the conflicting and non-conflicting case incorporate two separate constraints that need to be satisfied. In the case of the derivative constraint, we only have one constraint, whose violation is listed under Inf-1. Finally, for the bound constraint, the violation is averaged through the constraint domain points and listed under Inf-1. Figure 1 and Figure 2 show a visual comparison of these two approaches. BENN shows significant differences with PR-BNN in the way conflicting constraints are handled. As a result, PR-BNN biases its prediction towards one of the two value constraints, whereas the BENN prediction occurs in between the two specified value constraints. The addition of the variance constraints in this work also allows experts to express uncertainties in the region of the specified constraints. The results of the derivative and bound constraints, however, do not show significant differences. It is to be noted however, that the PR-BNN method does have lower constraint violations compared to the BENN.

Table 1. Comparison of BENN and PR-BNN methods for value, bound and derivative constraints

Experiment	BENN		PR-BNN	
	Inf-1	Inf-2	Inf-1	Inf-2
Value Constraints - No Conflict	0.023	0.041	0.001	0.038
Conflicting Value Constraint	1.017	0.982	0.001	2.001
Derivative Constraint	0.0016	-	4.216e-06	-
Bound Constraint	0.0	-	0.0	-

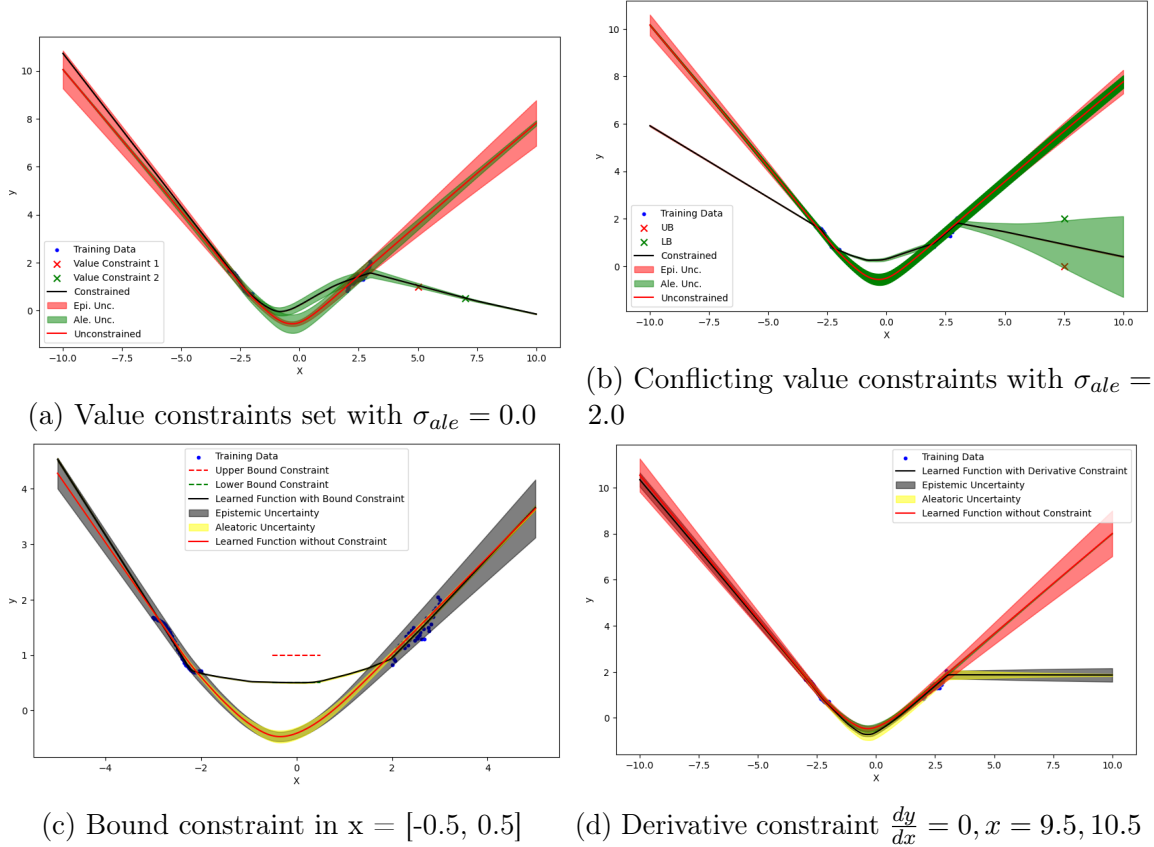


Figure 1. 1-D constrained regression demonstrations with value, derivative and bound constraints

3.3.2 Beam Deflection Problem

In this section, a classical problem in structural engineering is addressed: predicting the deflection of a beam under a given load. The data for this experiment is obtained

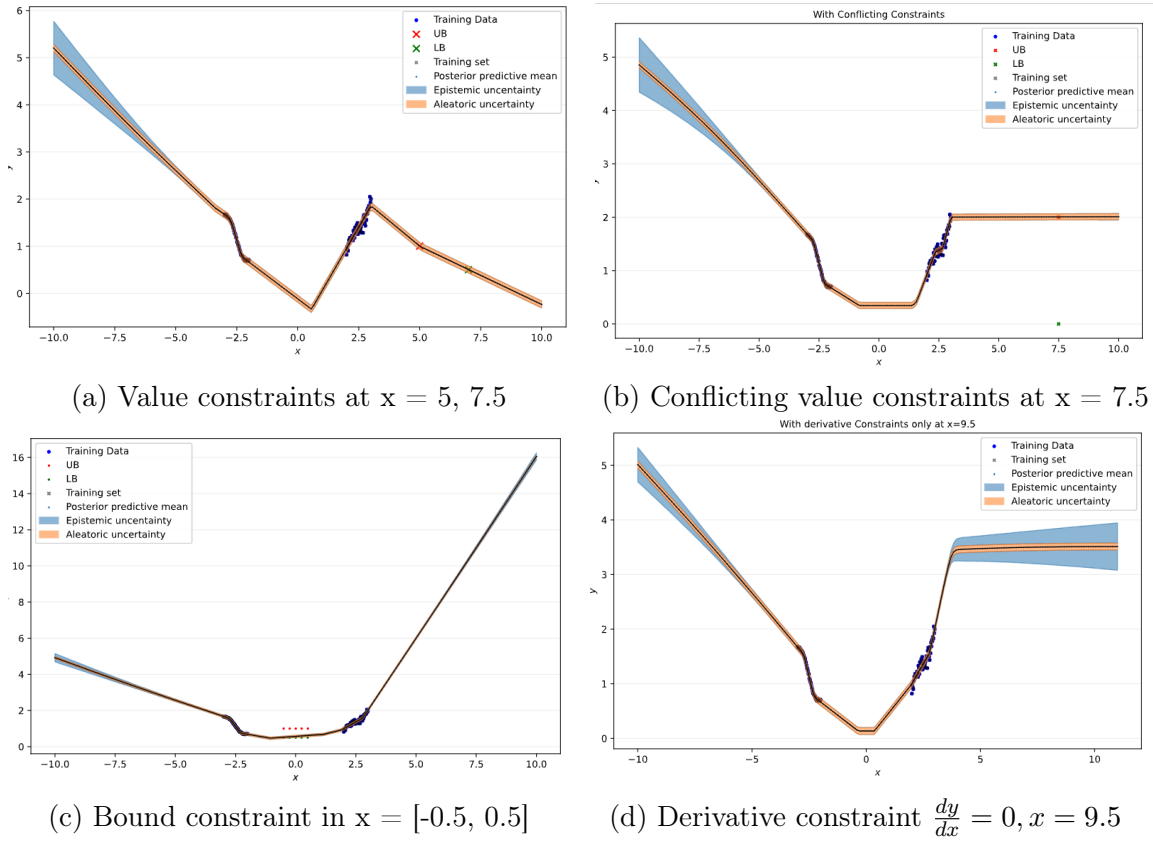
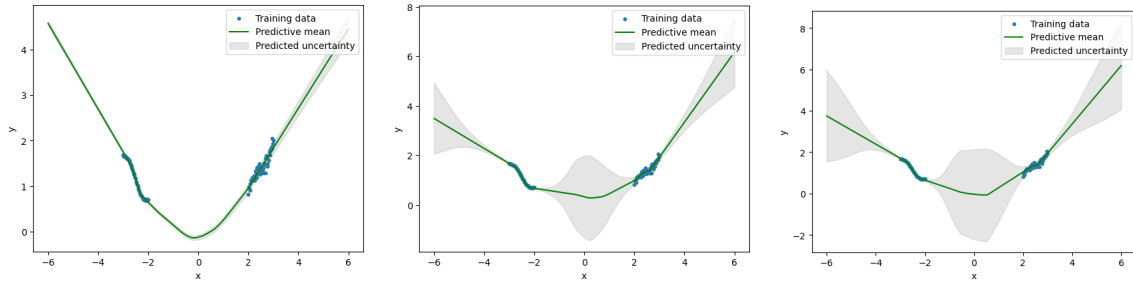


Figure 2. 1-D constrained regression demonstrations using PR-BNN with value, derivative and bound constraints



(a) Low variance outside the training data (b) Moderate variance outside the training data (c) High variance outside the training data

Figure 3. 1-D constrained regression demonstrations with variance constraints outside the training data

from the theoretical deflection of a beam subjected to a load. The deflection of a beam, y , as a function of its position, x , is given by the function $y(x)$, where the deflection depends on the material's Young's modulus (E), the length of the beam (L), the moment of inertia of the beam's cross-section (I), and the applied load (P). Specifically, the deflection function is defined piecewise for a beam of length $2L$ as:

$$y(x) = \begin{cases} \frac{-P}{8EI}x^3 + \frac{PL}{4EI}x^2, & \text{if } 0 \leq x \leq 2L \\ \frac{P}{6EI}x^3 - \frac{3PL}{2EI}x^2 + \frac{7PL^2}{2EI}x - \frac{7PL^3}{3EI}, & \text{otherwise} \end{cases} \quad (3.15)$$

In this experiment, the values for E , L , I , are set to typical values found in steel beams and P is set to 2000 N. For training purposes, a small samples of 10 points is observed by sampling points between $0.5L$ and $1.3L$. To simulate real-world measurements, a small Gaussian noise $N(0, 0.001)$ is added to the deflection values, resulting in the observed deflections, $y_{\text{train}}(x)$. The Bayesian Neural Network (BNN) used for regression predicts both the mean and the variance of the target variable. This network has an input layer with 1 neuron (representing the position x), and an output layer with 2 neurons, outputting both the predicted mean deflection and the log-transformed noise level. Further, it has 1 hidden layer, consisting of 2048 units. The GELU activation function $x * \Phi(x)$ is used for this task, where $\Phi(x)$ is the Cumulative Distribution Function (CDF) of the Gaussian. This activation helps improve performance in the mean sense for the test set, evaluated from $L = 0$ to $L = 3$.

3.3.2.0.1 Constraint Implementation

To enhance the performance of the BNN model, specific constraints are applied. These constraints are formulated to ensure that the model's predictions adhere to

known physical principles and boundary conditions associated with beam deflection. Two types of constraints are implemented: value constraints and derivative constraints.

1. **Value Constraint:** The value constraint is applied to ensure that the predicted deflection at specific points matches known values. This is particularly relevant at the beam's left end and the pin support at $x = 2.0$, where the deflection is zero:

$$f(x = 0.0; \theta) = 0, \quad f(x = 2.0; \theta) = 0. \quad (3.16)$$

2. **Derivative Constraint:** The derivative constraint is applied to ensure that the rate of change of the deflection at a certain point is known. In this case, the derivative is enforced to be zero at the point $x = 0.0$, corresponding to the beam's left end. The derivative constraint function approximates the derivative using the finite difference method between two closely spaced points, $x = 0.0$ and $x = 0.05$, and constrains this derivative to be zero. Mathematically, it is formulated as:

$$\left. \frac{df(x; \theta)}{dx} \right|_{x=0.0} = 0. \quad (3.17)$$

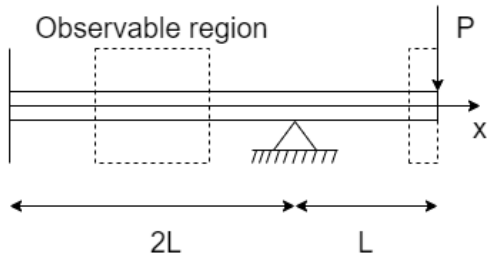


Figure 4. Beam configuration. The observable regions are given in the dashed boxes.

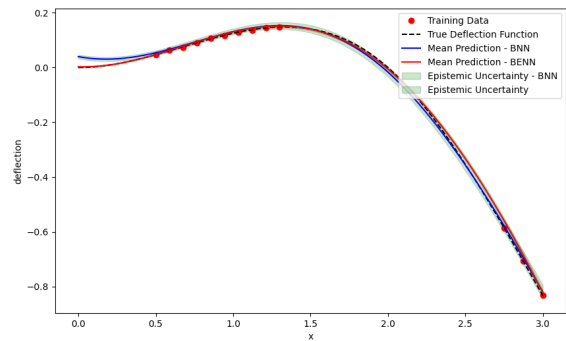


Figure 5. Predicted beam deflection with BENN and BNN.

Figure 5 shows that the proposed BENN model prediction significantly outperforms the BNN model.

3.3.3 Microstructure Generation

In materials science, predicting the properties of heterogeneous materials such as alloys, composites, polymers, and porous media are critical. These materials exhibit complex microstructures that influence their mechanical, thermal, electromagnetic, and chemical properties. Traditional experimental approaches to study these properties are often time-consuming and computationally intensive. Autoencoders, leveraging advancements in machine learning, provide a modern computational tool that offers significant benefits in the simulation and analysis of microstructures.

Autoencoders facilitate the fast and efficient simulation of microstructures, reducing reliance on extensive experimental datasets and accelerating the material design process. This capability is crucial for designing materials with optimized properties tailored for specific applications, essential in sectors like aerospace, automotive, and biomedical engineering. Furthermore, the computational generation of microstructures allows for the quantification and management of the inherent uncertainties in materials, which is vital for developing reliable materials and implementing robust design principles.

Moreover, generating microstructures using autoencoders fits seamlessly into broader computational frameworks such as Integrated Computational Materials Engineering (ICME), as discussed in (Gao *et al.*, 2021). (Gao *et al.*, 2021) proposes a method using a mixture random field model to generate a non-gaussian field. This statistical method can generate binary phase microstructures, that can either be anisotropic or isotropic, and requires the specification of two point correlation func-

tions. The proposed autoencoder-based approach is aimed at further increasing the computational speed of generating these microstructures.

3.3.3.1 Bayesian Entropy Convolutional Variational Auto-Encoder (BE-CVAE)

The proposed architecture integrates a Convolutional Variational Autoencoder (CVAE) focusing on the image generation while accommodating specific constraints using the BE framework. The encoder is a sequence of convolutional layers, each followed by a Rectified Linear Unit (ReLU) activation function, MaxPooling for spatial downsampling, and Batch Normalization. The convolutional layers extract hierarchical features to finally obtain the latent space Z .

The latent space is characterized by two fully connected layers, one for the mean μ and the other for the log-variance $\log(\sigma^2)$, of a Gaussian distribution. The model uses the reparameterization trick, generating a sample z from the latent space by where ϵ is a random noise sampled from a standard normal distribution.

The decoder reconstructs the input image using transposed convolutional layers with ReLU activations and Batch Normalization. A Sigmoid function in the final layer outputs pixel values in the $[0,1]$ range. Post-decoding, the output is resized to the original input dimensions using interpolation.

3.3.3.2 Dataset

The dataset is generated using a second-order non Gaussian random field, without using techniques such as simulated annealing, which are computationally intensive. The use of this technique allows for the simultaneous generation of both the microstructure

and the latent material property field. Using this approach, upto 150 training samples are generated for the microstructure generative model experiments.

3.3.3.3 Constraint - Two Point Correlation Function

The Two Point Correlation Function (TPCF) serves as a fundamental constraint in the analysis and synthesis of binary microstructure images. This statistical tool quantifies the spatial correlation between pairs of points within a given distance in a microstructure, providing insight into its heterogeneity and spatial distribution features.

In the proposed model, the TPCF is calculated for each generated microstructure image. The function evaluates the probability of finding two points, separated by a certain distance, that are both in the same state (either material or void). The TPCF constraint ensures that the generated microstructures possess similar spatial characteristics to the training set. This includes features like the distribution of phases, the degree of homogeneity or clustering, and the overall geometric properties of the material. By applying this constraint, it is aimed to produce microstructures that are not only statistically representative of the real material system but also maintain critical physical and mechanical properties.

Ensuring fidelity to the TPCF is particularly significant in materials science and engineering, where the microstructural arrangement greatly influences the macroscopic properties of materials. The accurate reproduction of TPCF in synthetic microstructures is thus essential for reliable material property predictions and subsequent applications.

The Two Point Correlation Function (TPCF) in this study is computed using a

Fourier Transform Autocorrelation approach, which is an efficient method for analyzing spatial patterns in microstructures. The steps of the computation, described in detail in (Suankulova, 2020), are summarized as follows:

1. **Fourier Transform:** The Fourier Transform of the binary microstructure is computed. This step is facilitated by the use of the Fast Fourier Transform (FFT), which is computationally efficient for digital images.
2. **Power Spectrum:** Next, the power spectrum of the Fourier Transformed image is obtained by calculating its squared magnitude. This power spectrum reflects the frequency components of the spatial pattern of the microstructure.
3. **Inverse Fourier Transform:** The inverse Fourier Transform is applied to the power spectrum. This operation converts the frequency-domain representation back into the spatial domain, resulting in an autocorrelation function.
4. **Radial Averaging:** Finally, for the TPCF, radial averaging is performed on the autocorrelation function. This process involves averaging the values over circles (in 2D) of radius r , which provides the final TPCF.

This Fourier Transform approach to computing the TPCF is advantageous due to its computational efficiency, particularly for large and complex microstructures. It also inherently accommodates periodic boundary conditions, which is beneficial in materials with repeating structures.

The resultant TPCF from this method provides insights into the heterogeneity and spatial distribution of features within the microstructure, crucial for understanding how these features influence the material’s macroscopic properties.

To facilitate gradient-based optimization in the process of image binarization, a differentiable binarization function is employed, defined as follows:

$$b(x) = \sigma(s \cdot (x - \theta)), \tag{3.18}$$

where σ denotes the sigmoid function, s is the steepness parameter, θ is the threshold, and x is the input value. The sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$ is used to provide a smooth transition between 0 and 1. The steepness parameter s , set to a high value (e.g., 100), ensures a sharp transition around the threshold θ . This approach allows for an approximate binary representation while retaining differentiability, which is essential for gradient-based optimization methods.

The TPCF is now ready to be used as a functional constraint. This amounts to either solving multiple value constraints for the TPCF sequentially, or to compute the overall absolute deviation from the expected curve at once. The latter option is computationally cheaper than evaluating multiple value constraints per iteration.

3.3.3.4 Constraint - Porosity

The porosity of a microstructure is a critical characteristic, quantifying the fraction of void space within the material. The function is defined as follows:

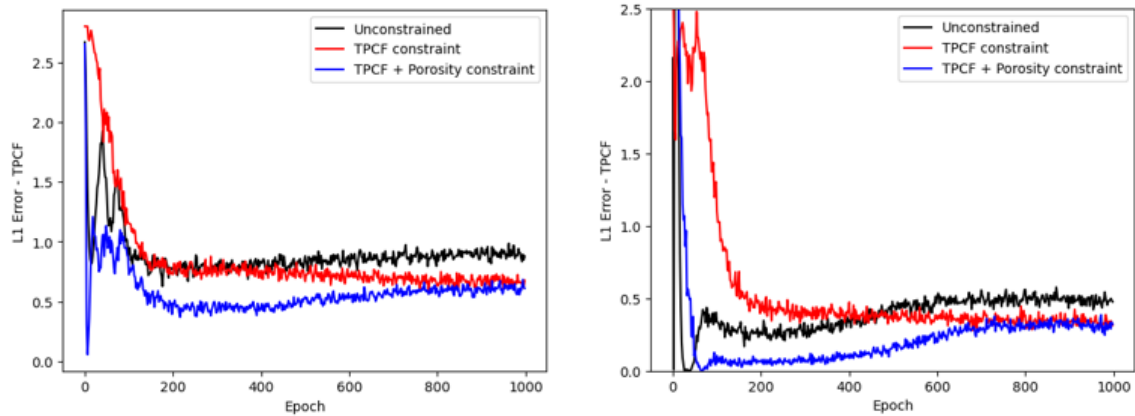
$$e = \frac{V_p}{V_p + V_s}, \quad (3.19)$$

where V_p is the total number of voxels in the void phase and V_s is the total number of voxels in the solid phase.

3.3.3.5 Results

The proposed model was first tested to compare how fast convergence to generating quality samples occurred. The baseline case with no constraints took far longer to converge compared to the ones using constraints. Furthermore, the model that used both porosity and the TPCF constraint converged significantly faster than both the

baseline and the model that only incorporated porosity constraints. Next, the effect of training data on performance is studied. The performance of the generative model is measured by how well the generated structures conform to the expected TPCF function. Table 2 shows strong evidence of improvement in the compliance with the TPCF requirements as more constraints are added and with more training data, and Figure 6 shows the L1 error of the TPCF with various types of constraints. It is to be noted however, that the evolution of constraint compliance during training paints a more nuanced picture in high dimensional problems. The resulting absolute errors from the TPCF and porosity constraints are lower those seen in the baseline model, it is still non-zero. This is because the TPCF and porosity constraints end up guiding the model to produce faster convergence with better reconstruction adherence with these two constraints.



(a) 25 training samples

(b) 150 training samples

Figure 6. Comparing the L1 error on TPCF across three microstructure generation scenarios: Unconstrained, TPCF constraint only, and TPCF + Porosity constraint

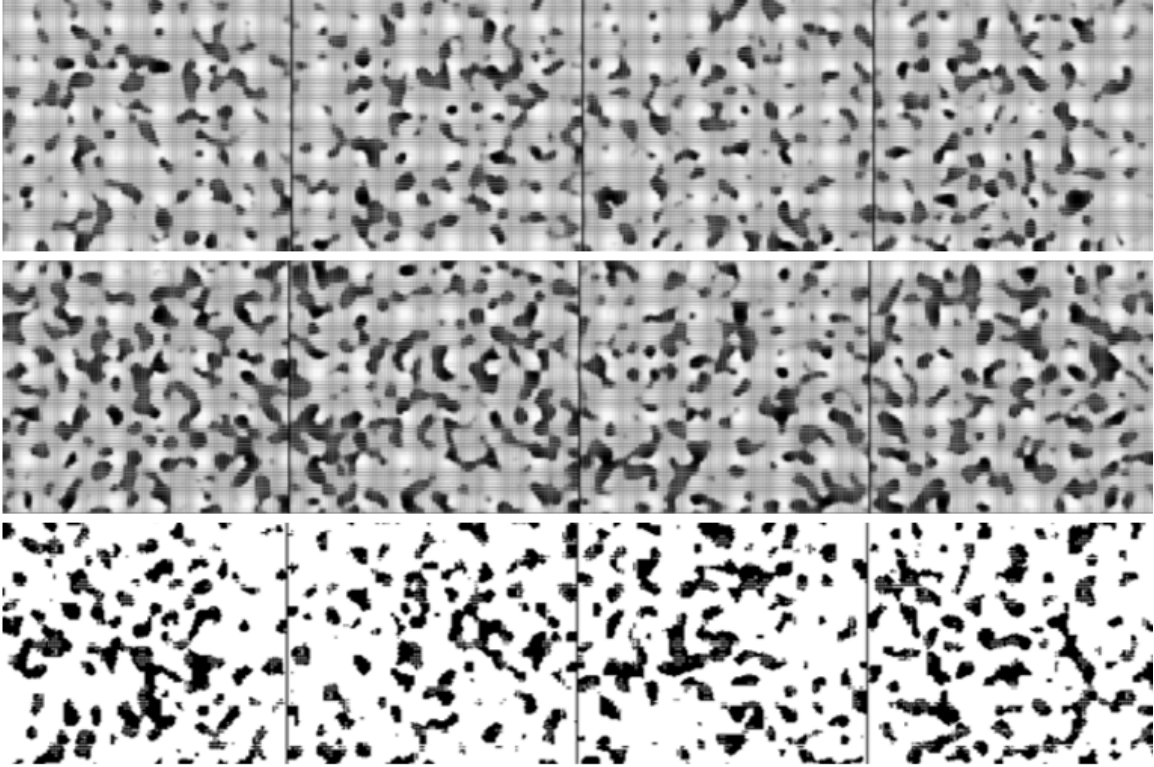


Figure 7. Microstructure generation using the proposed model after training the model with 260 epochs with (Top Row) No constraints (the baseline CVAE model) (Middle Row) TPCF Constraint (Bottom Row) TPCF + Porosity Constraints. Note that the model converges faster with two constraints compared to the baseline or just using the TPCF constraint

Training samples	No constraint	TPCF	TPCF + Porosity
25	0.85	0.67	0.60
50	0.59	0.46	0.38
100	0.59	0.41	0.39
150	0.47	0.29	0.34

Table 2. Constraint compliance as TPCF-L1 error on generated samples: Analysis along training data and number of constraints

3.4 Conclusion

In this chapter, the Bayesian Entropy Neural Networks (BENN) framework was introduced and its application in various settings, including 1D regression, beam

deflection modeling, and microstructure generation, was demonstrated. The integration of the Maximum Entropy (MaxEnt) principle with the Bayesian framework allowed for the imposition of constraints on predictions, thereby offering a novel approach to constrained deep learning. Note that this method is a general framework that explains the approaches used in similar works such as (Huang *et al.*, 2022) and (Yang *et al.*, 2020c).

The experiments with 1D regression emphasized the BENN framework’s versatility in handling value, derivative and variance constraints. It was observed that the model could effectively accommodate external constraints, even in areas with sparse or absent training data. This was particularly evident in experiments involving conflicting constraints and variance control, illustrating BENN’s ability to manage complex and uncertain information. The model can therefore be used for problems that require adherence to boundary conditions through constraints. An interesting demonstration of BENN in this work was in the domain of microstructure generation. The integration of constraints like the Two-Point Correlation Function (TPCF) and porosity into a Convolutional Variational Autoencoder showed substantial improvements in generating microstructures that are not only statistically representative but also improve adherence critical physical properties. This highlights the framework’s potential in materials science, where accurate microstructure characterization is crucial.

This chapter provides some areas for improvement and further investigation. The computational complexity associated with performing simultaneous optimization of model parameters and Lagrangian multipliers is a challenge, particularly for large-scale problems. Another avenue for advancement lies in the integration of epistemic uncertainty constraints within the BENN framework. Epistemic, or model, uncertainty arises from a lack of knowledge about the best model to represent a process. In

many real-world applications, especially in fields like climate modeling and epidemic modeling, understanding and quantifying epistemic uncertainty is crucial for making reliable predictions and decisions. Incorporating these constraints could enhance the model's ability to express uncertainty in predictions, especially in scenarios with limited data or where the data do not capture the entire spectrum of the underlying distribution. This could involve developing new methodologies for characterizing and quantifying epistemic uncertainty and devising novel ways to embed this information into the learning process. Finally, applying the BENN framework to a wider array of practical applications would not only demonstrate its versatility but also uncover specific challenges with this framework.

EPISTEMIC AND ALEATORIC UNCERTAINTY QUANTIFICATION FOR
CRACK DETECTION USING A BAYESIAN BOUNDARY AWARE
CONVOLUTIONAL NETWORK

4.1 Introduction

Cracks are a major source of failures in structural and industrial components. There are several reasons why cracks might form in these structures. For instance, roads and building cracks occur over time due to exposure to environmental factors such as humidity and rapid temperature changes. They also occur due to overload, such as heavy traffic in the case of roads. Phenomena such as corrosion can also lead to crack initiation and growth. These cracks, if undetected, can lead to catastrophic failure. Detecting cracks early is crucial to prevent such failures, improve maintenance protocols, reduce costs, and extend the lifespan of infrastructure.

Vision-based inspection offers a safe, efficient and rapidly scalable solution for crack detection. Other signal-processing modalities that are popular for crack detection include millimeter-wave imaging (Bivalkar *et al.*, 2022) and ultrasonic inspection (Sun *et al.*, 2023; Lee *et al.*, 2022). The primary benefit of vision-based inspection for crack detection is its versatility in various applications. It can detect various types of cracks depending on the sensor parameters, from hairline to large size, on a wide range of materials including composite, metal, ceramic, and plastic. Moreover, it is not affected by factors such as temperature and vibration. The primary drivers of improvements to industrial inspection technology using vision-based approaches

are higher computational power, miniaturized, commercially available sensors, and rapid advancements in machine learning techniques. These factors have opened up new possibilities for continuous condition monitoring of civil infrastructure at scale. In addition, autonomous inspection techniques provide an opportunity to remove subjectivity that can result from manual inspectors. While deep learning-based techniques are getting widely adopted for various types of industrial and infrastructure inspection tasks, for this study, the focus is narrowed down to working on crack detection in structural and infrastructural systems, such as crack detection in concrete surfaces and road pavements.

Accurately detecting cracks within an image is a complex and challenging problem. Convolutional neural networks (CNNs) have been used to recognize and classify cracks, but early deep learning-based models focused solely on the classification problem and ignore underlying feature information such as crack boundaries. Detecting accurate crack boundaries is important for downstream condition monitoring and maintenance scheduling. Traditional vision-based fault detection methods use hand-crafted features that limit generalization capacity. To address this, deep learning has been used to develop more flexible and accurate models for fault detections. Given that fine-grained crack detection provides us with highly resolved morphology in an end-to-end fashion, the question of whether the detection is to be trusted is important. This is an interesting problem because the adoption of deep learning approaches can be accelerated if one can quantify the confidence of the prediction produced by a neural network and address the problem of detecting distributional shifts that occur in practice after deployment. The consequences of poor detection results in a situation where the final risk assessment may be inaccurate, resulting in either a higher frequency of catastrophic events or maintenance cost overruns. Therefore, the first goal of this work

is to produce uncertainty estimates along with predictions for detection. To do this, a sampling-based Bayesian Deep Learning approach is used to decompose the sources of uncertainty in prediction, using the loss function proposed by (Kendall and Gal, 2017). Experiments to benchmark the validity of the proposed uncertainty quantification approach are done to ensure that the characteristics needed for detection are met. The next goal is to formulate the detection problem using a multi-task loss that learns a distributional loss using the log-likelihood term, refines the boundaries of the crack, and align it to the ground truth using a boundary loss term. This formulation borrows from Wang *et al.* (2021b) to compute a boundary loss that learns to better align predicted and ground truth boundaries. To demonstrate the effectiveness of the method, experiments are conducted on two commonly used crack segmentation datasets and report results on model performance, calibration, and uncertainty for both within and out of distribution samples.

The contributions of this work are as follows:

- A Bayesian Boundary Aware Convolutional Network (B-BACN) is proposed for crack segmentation that can predict aleatoric uncertainty, provide a sampling-based epistemic uncertainty, and refined boundary using the active boundary loss.
- Detailed empirical evaluations are conducted for within and out-of-distribution cases in order to analyze the effect that additional training samples have on uncertainty, predictive performance and model calibration.
- Analysis of the improvements that boundary losses can bring to improve the accuracy of the crack morphology detection is performed.
- Uncertainty and model calibration are used as an important performance index when assessing and comparing computer vision models in industrial settings,

where small dataset sizes and gradual distributional shift after deployment are commonly encountered.

The rest of the chapter is organized as follows: In Section 4.2, the literature on crack detection using image data is discussed. Following this, the development of uncertainty quantification techniques for condition monitoring models is discussed. After that, earlier work that used Bayesian approaches for uncertainty-aware defect detection is summarized. Section 4.3 introduces the problem setup and techniques used in this chapter for Bayesian inference when employing deep learning models and describes the proposed Bayesian Boundary-Aware Convolutional Network architecture. Section 4.4 outlines the implementation details for training the network, followed by a brief description of the evaluation metrics that are adopted. In Section 4.5, the proposed approach is evaluated, and it is seen that the boundary-aware defect detection provides significant performance improvements, while reducing uncertainty and improving calibration error. Detailed empirical evaluations using different loss weighting strategies, uncertainty quantification methods, and network architecture ablation studies are also provided. Finally, in Section 4.6, the key findings are summarized and a discussion on potential future work is included.

4.2 Related Work

4.2.1 Crack Detection Techniques

A lot of early studies that focus on the crack detection problem used methods that led to a series of improvements in detection capabilities which saw extensive feature engineering and signal processing efforts (Woods and Allen, 1989; Kirschke *et al.*,

1992; Mao-de *et al.*, 2007; Ayenu-Prah *et al.*, 2008). The availability of more imaging data and the explosion of deep learning led to bounding box approaches for detecting cracks, which were mostly inspired by the YOLO (Redmon *et al.*, 2016) and R-CNN (Girshick, 2015; Ren *et al.*, 2015). These works exploited pre-trained backbones from these networks and fine-tuned it on crack datasets (Deng *et al.*, 2021; Mao *et al.*, 2020; Hacıfendioglu and Basaga, 2021; Kato *et al.*, 2022). While these approaches effectively exploited the availability of higher compute power and pre-trained weights for effective fine-tuning, they did not provide end-to-end pixel-wise prediction for cracks. Fine-grained crack detection results are crucial for any detailed morphological analysis of the structures of interest, so a bounding-box approach to this problem is not sufficient. To address this, semantic segmentation approaches were proposed and evaluated on benchmark datasets, alleviating the need for hand-crafted feature engineering and classical signal processing. The Fully Convolutional Network has been an established technique for object detection and segmentation in medical imaging, introduced by Long *et al.* (2017). This approach saw rapid and extensive adoption in the industrial inspection field, with improvements to the segmentation accuracy by introducing widely used concepts such as feature pyramid hierarchies (Yang *et al.*, 2020a) and hierarchical feature learning (Zou *et al.*, 2018; Cheng and Zhou, 2021), loosely tied to the seminal works by He *et al.* (2016a); Ronneberger *et al.* (2015).

4.2.2 Defect Boundary Reproduction and Detection

Improving the reproduction and detection of boundaries of objects is challenging and relevant to the crack segmentation problem, as outlined in Section 4.1. Over the years, multiple different approaches have been proposed, including early traditional

methods such as active contours (Chan *et al.*, 2001). Early work in boundary refinement also included applying post-processing methods such as the Conditional Random Field (CRF) based method proposed by (Krähenbühl and Koltun, 2011) and image filtering to refine crudely localized crack boundaries and then classify them using classical ML models, as seen in Shi *et al.* (2016). Since crack pixels are similar to edges, early works that used filters were inspired by edge detectors. However, these methods did not have semantic knowledge of the crack pixels and their relation to the background. This meant that a lot of post-processing and advanced filter design is required to remove false positives. Later works began to utilize approaches that defined boundary refinement blocks Chen *et al.* (2020). (Guo *et al.*, 2021a) adapts the original image gradient with the coarse crack detection result and refines it to precise crack boundaries.

4.2.3 Uncertainty Quantification for Reliable Condition Monitoring

Estimating model uncertainty remains a significant topic of interest in a wide array of applications such as condition monitoring (Moradi *et al.*, 2022; Seites-Rundlett *et al.*, 2021), fault diagnostics (Zhou *et al.*, 2022), and remaining useful life estimates (Zhu *et al.*, 2022). A lot of the works in reliability engineering focuses on utilizing predictive models to perform system-wide reliability assessments. (Moradi *et al.*, 2022) uses a Bayesian Network to model a system-wide network and analyze its component risks using a data-driven framework. After the deployment of a predictive model, it is vulnerable to distribution shifts over time. (Zhou *et al.*, 2022) proposed a Bayesian Deep Learning technique to structural component health using 1-D signals, considering potential OOD samples. Uncertainty Quantification (UQ) is used as a way to classify

these samples. Uncertainties have also been to query for informative samples for active learning techniques (Yang and Loog, 2016; Wang *et al.*, 2018). (Zhu *et al.*, 2022) uses this technique to predict battery degradation. Uncertainty quantification under distributional shifts for fault diagnostics has previously been studied by (Te Han, 2022) using an ensemble of neural networks, where the authors use a thresholding approach to determine whether the uncertainty associated with a prediction makes it more likely for the sample to have originated from outside the training distribution. Sajedi and Liang (2020) incorporated uncertainty into their structural health monitoring model by using Monte-Carlo Dropout sampling and prediction quality classification and showed that variance of softmax and entropy correlate with misclassification rate. Well calibrated uncertainties are key for condition monitoring, especially when using neural networks, which are notorious for providing overconfident estimates. Improving calibration has been studied by (Guo *et al.*, 2017), where the authors propose temperature scaling (TS) to tune the outputs of a neural network. Obtaining calibrated predictions for regression tasks using neural networks has been studied by Tohme *et al.* (2022), where the authors propose a novel loss function formulation.

4.2.4 Bayesian Approaches for Defect Detection

While novel neural network architectures have improved model performance for crack detection, uncertainty estimates for crack detection has not been studied to the same extent. Deep learning techniques do not lend themselves to analytical Bayesian inference and require approximate inference techniques, which have seen a lot of development in computational statistics literature (Blei *et al.*, 2017; Zhang *et al.*, 2018). Among Bayesian Deep Learning methods, there have been a multitude of approximate

evaluation methods proposed, including Variational Bayes-By-Backprop (Blundell *et al.*, 2015), Monte Carlo Dropout (Gal and Ghahramani, 2016), Spectral-Normalized Gaussian Process (SNGP)(Liu *et al.*, 2020a). These approaches have been adopted in a variety of applications (Pang *et al.*, 2022, 2021; Lee *et al.*, 2017; Fan *et al.*, 2023) to estimate epistemic uncertainty. However, the quantification and decomposition of both epistemic and aleatoric uncertainty in the defect characterization context has been missing, with more attention being paid in the field to collecting domain-specific datasets and improving predictive performance. Therefore, this work analyzes the decomposition of epistemic and aleatoric uncertainties for crack characterization, and uses the uncertainty to assess model performance and calibration.

The study of uncertainty decomposition has recently been discussed by McFarland and DeCarlo (2020) in the context of risk assessment. McFarland and DeCarlo (2020) have focused on decomposing the epistemic and aleatoric components for problems such as cracking defect failure probability using classical Bayesian techniques. In order to extend the literature on uncertainty for defect characterization, this work aims to use the dropout approximation to Bayesian inference in neural networks to extract epistemic uncertainty (Srivastava *et al.*, 2014; Gal and Ghahramani, 2016) and a negative log likelihood formulation to predict aleatoric uncertainty. A recent work similar to ours on the uncertainty quantification front is (Pyle *et al.*, 2022), where they provide uncertainty analyses for crack characterization using ultrasonic sensors for in-line inspection. (Pyle *et al.*, 2022) studies uncertainty on a specialized ultrasonic crack characterization dataset, and comments on uncertainty analyses using various quantification methods for out-of-distribution (OOD) cases. However, this study focuses on using uncertainty to analyze improvements to model performance

and calibration on OOD and within-distribution data using a boundary-refinement objective while training the model.

In summary, the crack detection model combines uncertainty decomposition and boundary refinement techniques to produce detection results that result in two main benefits: the ability to accurately segment crack morphology, and the ability to separately compute epistemic and aleatoric variances. The uncertainty is used to analyze prediction confidence and whether the model is well-trained for the test set being considered. This work also analyzes the effect that additional training samples have on uncertainty, and provides detailed empirical evaluations of within and out-of-distribution cases.

4.3 Methodology

The overall architecture is based on the fully-convolutional encoder-decoder network (Long *et al.*, 2017), with a Resnet-50 backbone, as shown in Figure 8. The encoder network consists of 5 blocks, with pre-trained weights from the ImageNet dataset that are retrained for each training run. The encoder blocks provide a hierarchy of features at multiple scales, with earlier layers extracting fine-grained morphological information, and later layers extracting coarse-grained category and location information. The resulting layers are then passed into the decoder, which consists of transposed convolutions such that the feature outputs mirror the corresponding size of the next encoder layer. In addition to having information flow sequentially through each layer in the network, the skip-connections across the encoder-decoder structure combines the representations obtained across multiple scales. This approach

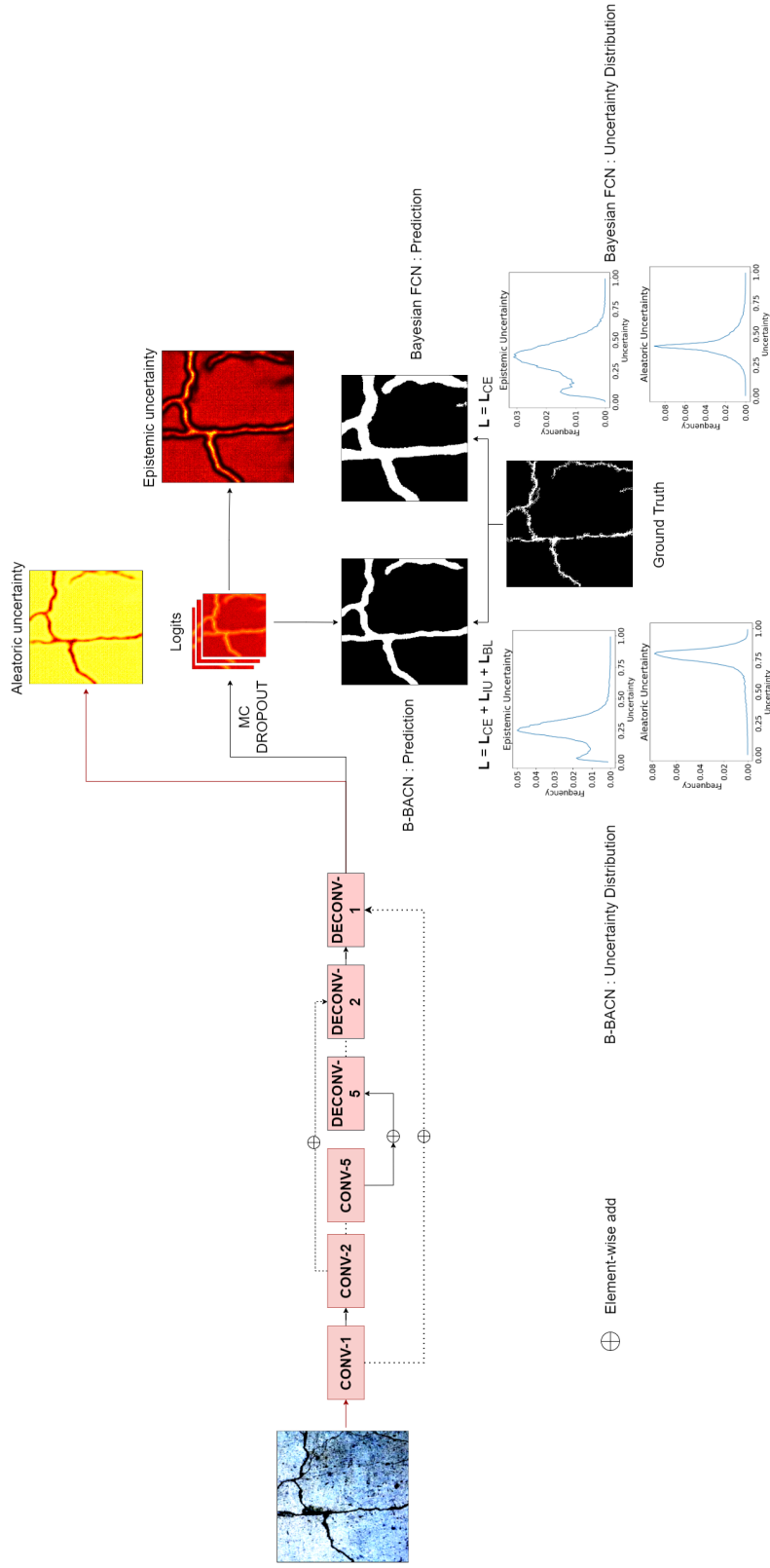


Figure 8. Overview of the proposed network architecture showing the network structure, input, predictions and uncertainty. The use of the proposed approach allows the model to predict tighter crack boundaries and epistemic uncertainties.

has been successfully used in the past to help with boosting gradient flow, improving the vanishing gradient problem.

Structural prognostics benefit immensely from being able to model predictive uncertainty at multiple stages of analysis. Point estimates provided by deep neural networks may lead to overconfident predictions that can often be wrong. Expressive models provide for downstream analysis and have a way to explain its predictions to the end user are highly sought after.

Bayesian methods are considered in this study, with a major focus on MC Dropout (MCD) with a predictive aleatoric variance output learned during training. Concrete Dropout (CD) (Gal *et al.*, 2017) and Bayes By Backprop (BBB) (Blundell *et al.*, 2015) are compared against this method. Note that the MCD approximation is shown to be a special case of the generic VI approximation in Gal and Ghahramani (2016). The question here is whether the uncertainty is purely derived from the uncertainty about the weights. The analysis of the variance by Brach *et al.* (2020) provides some insight into this problem:

$$\begin{aligned}
 E_i^D &= E_i(1 - p^*) \\
 V_i^D &= V_i p^*(1 - p^*) + V_i(1 - p^*)^2 + E_i^2 p^*(1 - p^*)
 \end{aligned}
 \tag{4.1}$$

Equation 4.1 shows that the expectation and variance of the output E_i^D and V_i^D only depend on the expectation E_i and variance V_i from the previous layer and the applied dropout ratio p . This shows that the only contribution to the variance in MC Dropout is from the dropout layer. Therefore, this component of uncertainty is driven by the uncertainty in the weights.

It is also important to look at the effect of the dropout objective when applied to learning problems. For example, in the case of regression, dropout regularization is equivalent in expectation to the ridge regression model. The input feature X

$\epsilon R^{N \times D}$ is to be transformed to a target $Y \in R^{N \times 1}$, using weights $w \in R^{D \times 1}$. Applying dropout implies multiplying the feature element-wise with a dropout mask matrix $Z \sim \text{Bernoulli}(p)$. Casting this with dropout applied to the feature in the mean-squared error loss gives:

$$L(w) = \underset{w}{\text{argmin}} E_{(Z \sim \text{Bern}(p))} \|y - (Z \odot X)w\|^2 \quad (4.2)$$

Computing the expectation of the squared error loss gives us the following:

$$L(w) = \|y - X\tilde{w}\|^2 + \frac{1-p}{p} \|\Gamma\tilde{w}\|^2, \tilde{w} = p.w \quad (4.3)$$

Directly comparing the loss for the dropout with the L2 regularized loss shows that it has stronger restrictions on its Lagrange multiplier term. Concretely, $\lambda = \frac{1-p}{p}$ for dropout, where p is the dropout probability hyperparameter, compared to the looser restriction on the multiplier for the ridge loss, $\lambda \geq 0$.

Classical Bayesian linear regression formulations as seen in Bishop and Nasrabadi (2006) assume a constant, known value for the aleatoric uncertainty term. However, this assumption does not hold for real-world data, as some training samples can have higher variance than others. To remedy this, the proposed model learns a component of the variance as a function of the data, referred to as the heteroscedastic aleatoric uncertainty - the component of uncertainty that cannot be explained away with more data. The epistemic component of the uncertainty is calculated using Monte Carlo (MC) dropout during inference. In the context of applying uncertainty quantification to detection, epistemic uncertainty plays a more important role than aleatoric uncertainty, unlike in natural image situations with a lot of data, where the aleatoric uncertainty matters more. Sources of variability such as occlusion can be considered “input-dependent” and are captured by the heteroscedastic aleatoric uncertainty term. On the other hand, the industrial imaging domain does not always

have a lot of samples to train a model on, which requires epistemic uncertainty to reflect this. The approach used by (Kendall and Gal, 2017) distinguishes the epistemic and aleatoric uncertainty by deriving the loss function from the likelihood formulation. This is first demonstrated on a toy classification problem.

Epistemic and aleatoric uncertainty are related to the decision boundary in classification. This is visualized in Figure 9, a low dimensional example. The two moons dataset is a non-linearly separable dataset with 2 inputs. This low dimensional toy problem can be demonstrated with a large set of training samples to clarify the role of epistemic and aleatoric uncertainty. Since the model has 2 categories, the network outputs 4 values, 2 for the prediction logits and 2 for each of the variances. The variance terms are the input-dependent uncertainties. For a classifier:

$$y_i|x_i \sim \text{Bern}(\psi(w^T x_i)) \quad (4.4)$$

Assuming softmax activation ψ , the optimal weights can be derived using Maximum Likelihood Estimation (MLE):

$$w = \text{argmax}_w p(y_i, x_i|w) \quad (4.5a)$$

$$w = \text{argmax}_w p(y_i|x_i, w)p(x_i|w) \quad (4.5b)$$

$$w = \text{argmax}_w p(y_i|x_i, w) \quad (4.5c)$$

Equations (4.5a)-(4.5c) lead to a stochastic negative log-likelihood loss by first setting up a Gaussian distribution to sample from. The mean of this distribution is taken as the prediction $f(x_i)^w$. The variance term comes from the learned aleatoric variance output from the network. One can then sample t times from this distribution and then average its log-softmax over the sampling dimension. To avoid underflow and overflow issues, this is implemented by first transforming the $y_{i,t}$ using the log sum exp trick and then averaging over the sampling dimension. The final form of the loss

is implemented directly using the *Negative Log-Likelihood (NLL)* loss, after obtaining the samples as described, and comparing the prediction against the ground truth. The loss is stochastic because it depends on Monte Carlo draws from the variance term.

$$\begin{aligned}
 y_i|w &\sim N(f(x_i)^w, \sigma_i^{w^2}) \\
 y_{i,t} &= f(x_i)^w + \epsilon_t, \epsilon_t \sim N(0, \sigma_i^{w^2}) \\
 \mathcal{L}_{MLE} &= NLL(y_i, y_{gt})
 \end{aligned}
 \tag{4.6}$$

During the model evaluation, the procedure to compute uncertainty is by doing Monte Carlo sampling using dropout to obtain predictions. The epistemic uncertainty is computed as the variance of these predictions. The aleatoric uncertainty is the learned component of the prediction and is directly obtained from the formulation. Figure 10 shows that the epistemic uncertainty reduces with more training data while the aleatoric uncertainty remains oscillatory and independent of this trend. Figure 9 shows a demonstrative example of what the decision boundary looks like when using MC-Dropout. The areas with more data have a lower uncertainty in the decision boundary, indicated by the lower amount of variance in classification. At the extreme ends of the decision boundary, a widening of the noise profile is seen, indicating a higher uncertainty for the discriminative model to assign a fixed label to a sample near the boundary.

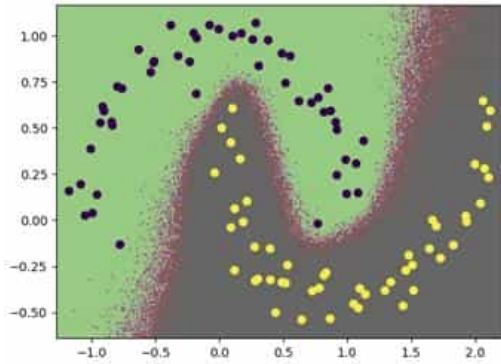


Figure 9. Classification with uncertainty in the two moons dataset - Variation of uncertainty and entropy with sample size

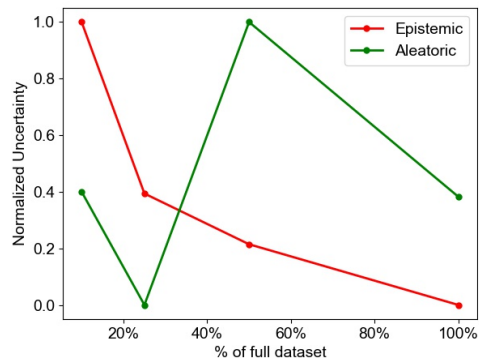


Figure 10. Classification with uncertainty in the two moons dataset - Variation of uncertainty with sample size.

The next component is the active boundary loss (ABL) that further refines

predictions, proposed by (Wang *et al.*, 2021b). This loss improves the alignment between predicted boundaries (PDBs) and ground-truth boundaries (GTBs) during training by moving PDBs toward the closest GTBs. The ABL loss is differentiable and dynamic and it focuses on the relationship between PDB and GTB pixels. It can be used in combination with other loss terms such as cross-entropy loss and Lovasz-softmax loss (Berman *et al.*, 2017) to improve the boundary details in image segmentation. The method can be useful in preserving the boundaries of thin objects in an image. The ABL continuously monitors changes in the PDBs to determine the plausible moving directions. The method is divided into two phases: First, for each pixel i on the PDBs, the next candidate boundary pixel j closest to the GTBs is determined. Second, the KL divergence is used to encourage the increase in KL divergence between the class probability distribution of i and j . Meanwhile, this process reduces the KL divergence between i and the rest of its neighboring pixels. In this way, the PDBs can be gradually pushed toward the GTBs. The ABL is determined as a weighted cross-entropy loss over the boundary pixels numbering N_b between the PDB direction D_i^p and the GTB direction D_i^g , with the weight w_i being the closest distance of the current PDB pixel to the GTB.

$$L_{ABL} = \frac{1}{N_b} \sum_{i=1}^{N_b} w_i \cdot \text{CE}(D_i^p, D_i^g) \quad (4.7)$$

Unfortunately, there are sources of performance degradation such as conflicts in candidate boundary pixels, so the authors use gradient flow control to reduce the effect of this. Further details of this process are elaborated in (Wang *et al.*, 2021b).

Additionally, a continuous, differentiable surrogate for the Jaccard index is used, given by the Lovasz-softmax function (Berman *et al.*, 2017), to refine the regional shape of the detected crack by directly optimizing for higher Intersection-over-Union (IoU) score. This loss provides for a way to learn by computing a differentiable version

of the IoU score. The original IoU score is a mapping from a binary N-Dimensional matrix to a value in the real number space, $f : \{0, 1\}^N \rightarrow R$. The modification of the domain to $[0, 1]^N$ is given by the Lovasz extension (Berman *et al.*, 2018), given by:

$$\bar{\Delta} : \mathbf{m} = \sum_{i=1}^p m_i g_i(\mathbf{m}) \quad (4.8)$$

In the multiclass setting, the outputs are first transformed into probabilities using the softmax, and then the vector of probabilities per class are used to compute an error vector. This error vector is used as a surrogate to compute the Lovasz extension of the IoU.

$$f_i(c) = \frac{\exp(l_i)}{\sum_K \exp(l_k)} \quad (4.9)$$

$$m_i(c) = |Y_i - f_i(c)| \quad (4.10)$$

$$L_{IU} = \bar{\Delta} J_c(m(c)) \quad (4.11)$$

To backpropagate, the errors computed in Equation 4.10 are sorted in decreasing order and the gradients are computed using the sorted $m(c)$. To compute the gradient, the intersection and union parts of the IoU are computed as described in Algorithm 1 of (Berman *et al.*, 2018).

Effectively, the IoU loss ensures that small defects are captured, and the predicted boundaries of these defects are optimized by the ABL loss. The total loss is computed as the combination of the supervised and consistency losses, with a sigmoid ramp-up weight function for the consistency loss such that it becomes more dominant in later epochs.

An important component of model calibration is to ensure that the model is not overconfident in its predictions. To attempt to improve this, temperature scaling is performed on the logits to calibrate the predicted probability distributions of

the network, following Guo *et al.* (2017), where the temperature is a learnable hyperparameter (T). Temperature scaling is commonly used to calibrate the confidence of the model, by scaling the logits of the mean prediction head. The temperature scaling procedure is summarized as follows:

1. Obtain the pre-softmax outputs logits of the trained neural network for the validation set.
2. Iterate over a range of temperature scalar values and select the one that minimizes the NLL loss over the validation set using the L-BFGS optimizer.
3. Divide the logits by the optimal temperature scalar.

In summary, the overall proposed loss is the combination of the derived heteroscedastic classification loss, the IoU loss and the ABL loss:

$$L = L_{MLE} + L_{ABL} + L_{IOU} \quad (4.12)$$

4.4 Experimental Setup

4.4.1 Implementation Details

The experiments are performed on the CrackForest dataset by Shi *et al.* (2016) and DeepCrack dataset (Liu *et al.*, 2019c). The CrackForest dataset is representative of a variety of cracks encountered in urban roads, and has been used as a benchmark for multiple studies such as Shi *et al.* (2016), Fan *et al.* (2018), Yang *et al.* (2019), Zou *et al.* (2018). It consists of 100 images, with 18 left out for the test set. The DeepCrack dataset is a larger dataset consisting of crack images at multiple scales and background textures. It consists of 443 training set images and 78 test set images.

The network architecture used for the main experiments is the Resnet50 backbone encoder with a transposed convolutional decoder, as explained in Section 4.3, and the models are trained using PyTorch. The training code has the ability to run in both single and multi GPU configurations. The encoder uses a set of pre-trained weights from the ImageNet dataset, which is then fully retrained using the specific dataset for the experiment. Optimization is performed using the SGD with momentum optimizer, with a momentum value of 0.9 and weight decay of 10^{-5} , along with a step-decay learning rate scheduler across all experiments. The learning rate is initialized at 0.01, which decays every 50 epochs by a factor of 0.8. During training, early stopping based on the validation loss history over the past 50 epochs is utilized. The early stop condition used for the experiments is an average validation loss reduction of less than 0.001. For the experiments across training samples, a dropout probability of 0.5 was used. The epistemic uncertainty during model evaluation is calculated using 25 MC runs through the network.

As part of these experiments, a study on how the weighting of the three losses influences performance is also included. Specifically, three strategies are considered:

1. Simple linear superposition (Baseline): Here, all three losses are directly added up, as seen in Equation 4.12
2. Ramp-up boundary loss weighting (Ramp): In this case, the boundary loss terms are weighted using a sigmoid ramp-up function. The ramp-up ends after 100 epochs, determined empirically based on the loss convergence of the cross-entropy loss:

$$L = L_{MLE} + \phi(t)(L_{ABL} + L_{IOU}), \quad (4.13)$$

where ϕ is the sigmoid function, and t is the epoch number.

3. Coefficient of Variations Weighting (CoV) (Groenendijk *et al.*, 2020): The method is founded on the Coefficient of Variation, which is the ratio of the standard deviation to the mean and shows the extent of variability of the observed losses in relation to their mean:

$$L = \alpha_{1t}L_{MLE} + \alpha_{2t}L_{ABL} + \alpha_{3t}L_{IOU} \quad (4.14a)$$

$$\alpha_{it} = \frac{c_{lit}}{z_t} = \frac{1}{z_t} \frac{\sigma_{lit}}{\mu_{lit}} \quad (4.14b)$$

$$l_{it} = \frac{L_t}{\mu_{L_{t-1}}} \quad (4.14c)$$

Equation 4.14a represents the weighted total loss function, composed of three terms weighted by $\alpha_{1t}, \alpha_{2t}, \alpha_{3t}$. Equation 4.14b defines the weighting term α_{it} for each component, defined as the coefficient of variation of the loss ratio l_{it} . The loss ratio l_{it} , found to be a robust training strategy by Groenendijk *et al.* (2020), measures the ratio of the current loss value to the mean of the loss history $\mu_{L_{t-1}}$, as seen in Equation 4.14c.

Table 3. Architecture details of the segmentation network

Layer	Filters	Filter size	Stride	Padding	Activation
ResNet-50	-	-	-	-	-
Conv2d	1024	3	1	1	ReLU
BatchNorm2d	1024	-	-	-	-
ConvTranspose2d	512	3	2	1	ReLU
BatchNorm2d	512	-	-	-	-
ConvTranspose2d	256	3	2	1	ReLU
BatchNorm2d	256	-	-	-	-
ConvTranspose2d	128	3	2	1	ReLU
BatchNorm2d	128	-	-	-	-
Dropout	-	-	-	-	-
Conv2d	-	1	-	-	-

Table 4. Training details and hyperparameters

Hyperparameter	Value
Optimizer	SGD with momentum
Momentum	0.9
Initial learning rate	0.01
Learning rate scheduler	Step Decay Learning Rate
Learning rate epoch decay interval	50
Decay factor	0.8
Early stop criterion	validation loss
Early stop - Loss history	50 epochs
Early stop - Average loss reduction	0.001

4.4.2 Evaluation Metrics

To quantify model performance, the macro F-1 score is used to report the mean F-1 score, which is an unweighted average of the class-wise F-1 scores. The F-1 score is defined as the harmonic mean of precision and recall and typically used to measure how well the segmentation captures crack morphology.

The equation for the F-1 score is:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \quad (4.15)$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad (4.16)$$

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (4.17)$$

The entropy and variance of the predictions over the MC runs are also computed to assess the level of uncertainty in the model out of those samples. The entropy is computed per-class by averaging the Monte Carlo mean prediction over each pixel as:

$$H(y) = - \sum_{c=1}^C y_c \cdot \log y_c \quad (4.18)$$

where C is the number of classes and y_i is the logit output of class i . The epistemic variance is computed across the set of MC predictions:

$$Var(y) = \frac{1}{M} \sum_{i=1}^M (y_i - \bar{y})^2 \quad (4.19)$$

where M is the number of MC samples, y_i is the sampled output from the network, and \bar{y} is the mean prediction.

Model calibration can be assessed using the Expected Calibration Error (ECE) (Guo *et al.*, 2017) (Naeini *et al.*, 2015) score. The ECE is computed using Equation 4.20, with the calibration bin accuracy $Acc(B_m)$ and the overall accuracy of the class m .

$$ECE = \sum_{m=1}^M \frac{|B_m|}{n} |acc(B_m) - conf(B_m)| \quad (4.20)$$

4.5 Results and Discussion

In the following experiments compare the baseline FCN model (Long *et al.*, 2017), the B-BACN model and the Temperature-Scaled B-BACN model on the DeepCrack dataset and the CrackForest datasets, establishing that the proposed B-BACN model significantly improves the detection performance, while reducing uncertainty and improving the expected calibration error. The technique used for modeling uncertainty, as demonstrated in Section 4.3, aims to capture specific behaviors of uncertainty, such as detecting higher uncertainty when the test samples are farther from the training samples and having an invariant aleatoric component with respect to the training samples. Therefore, the model performance across training samples and dropout is analyzed, revealing the utility of uncertainty metrics for determining whether the model is well-trained and to what extent these trends seen on the simpler two-moons dataset hold for the segmentation dataset. Next, a demonstration of how

the uncertainty model generalizes to unseen and out-of-distribution data samples from the CrackForest dataset is shown. Since the DeepCrack dataset is out of the distribution of the CrackForest dataset, it is expected that the epistemic entropy increases for these samples. Additionally, the aleatoric component of uncertainty is shown to not decrease despite training on the CrackForest dataset. Model performance is also demonstrated on smaller CrackForest dataset trained with noisy data, with and without the introduction of the boundary loss.

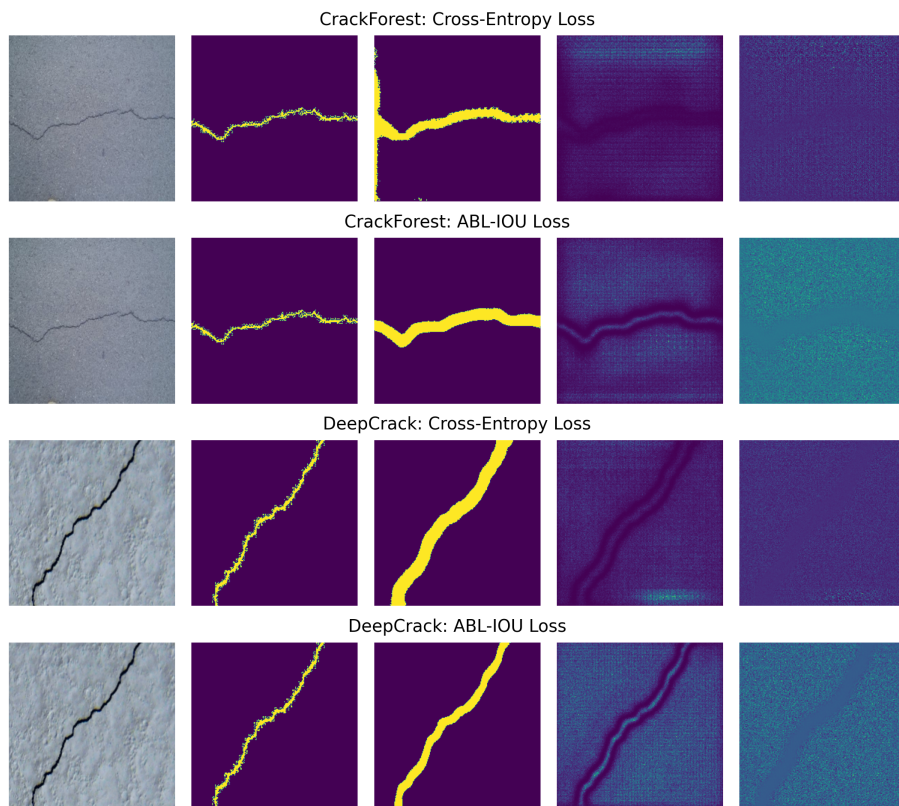


Figure 11. Demonstrative examples of detections in the CrackForest (Top-2 rows) and the DeepCrack Dataset (Bottom-2 rows). Images from Left to Right: Input, Prediction, Ground Truth, Epistemic Uncertainty, Aleatoric Uncertainty

The first set of results, shown in Table 5, compares model performance on the DeepCrack test set using the boundary loss against the MC Dropout baseline. Not

Table 5. Performance comparison for models trained on DeepCrack and CrackForest dataset

Dataset	Model	F1-score	Epistemic	Entropy	Aleatoric	ECE
DeepCrack	FCN	0.70890	0.00032	0.10525	1.02133	0.14657
DeepCrack	B-BACN	0.80060	0.00019	0.08406	1.03108	0.10371
DeepCrack	TS B-BACN	0.79581	0.00020	0.13612	1.03223	0.09010
CFD	FCN	0.63862	0.00054	0.13309	1.03619	0.09943
CFD	B-BACN	0.69882	0.00022	0.08280	1.01341	0.07617
CFD	TS B-BACN	0.69690	0.00025	0.14228	1.01340	0.05900

only does the boundary loss exceed F1 score compared to the baseline, but also shows a drop in the variance of the prediction. The models used were trained on the DeepCrack dataset using all the available training data. These results indicate that using the boundary loss terms helps to directly operate on and refine the pixel-level predictions. Temperature scaling enhances the Expected Calibration Error (ECE) without altering the predictions themselves. It achieves this by scaling each logit individually, while preserving the maximum logit as the maximum value. This is why there is no significant change in the F-1 score or the epistemic uncertainty. Similar trends are observed for the CrackForest dataset.

Figure 11 provides some demonstrative detection examples that compare these results. The predicted boundary changes were also compared between the boundary loss model and the cross entropy model, with the boundary loss model providing more accurate boundaries and lower uncertainty compared to the baseline.

Evaluating the model across various training samples shows a trend similar to the one seen in the toy problem in Section 4.3. These results are plotted in in Figure 12 and Figure 13. The epistemic uncertainty showed a rather jagged trend for the baseline cross-entropy trained model, but showed a stronger decreasing trend for

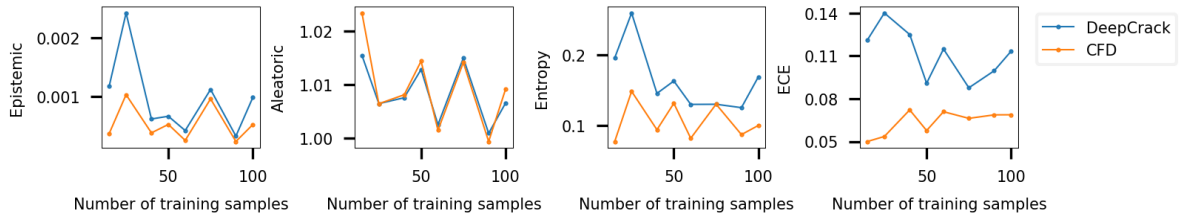


Figure 12. Evaluation of models trained on the CFD dataset across training samples on both the CFD test set and the DeepCrack test set.

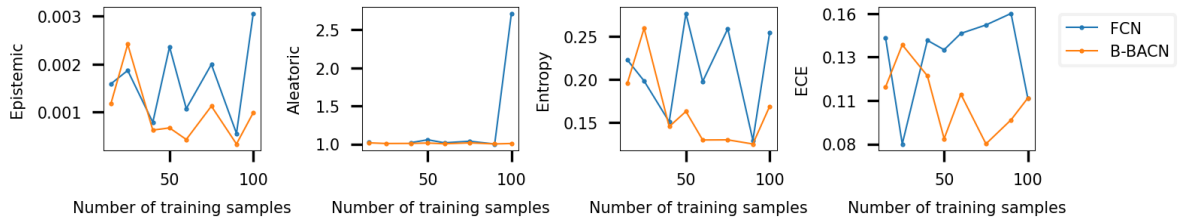
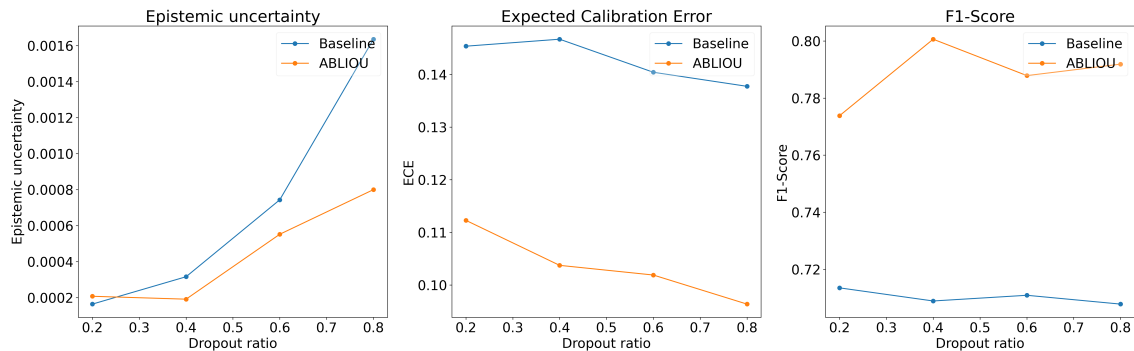


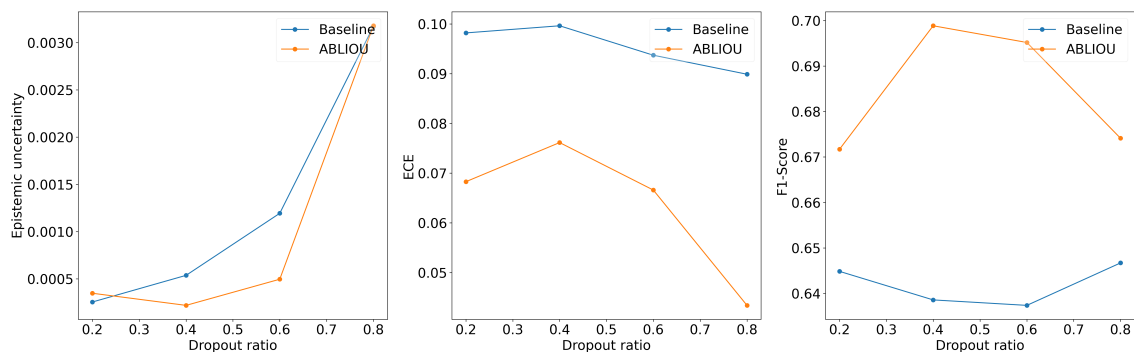
Figure 13. Evaluation of models trained on the CFD dataset across training samples on the DeepCrack test set on the Cross-Entropy trained Bayesian FCN baseline and the proposed B-BACN model.

the B-BACN. This result was somewhat surprising as one expects there to be an approximately inverse relationship between the amount of training data and the uncertainty. Given enough training samples, more information does not meaningfully change the uncertainty. (Chen *et al.*, 2010) shows that the result holds true only for normally distributed variables, and that the relationship between the uncertainty and information is not straightforward in other cases. The level of uncertainty is higher on the out-of-distribution DeepCrack dataset while the aleatoric uncertainty, as expected, does not change much. The calibration error is also higher in the out-of-distribution dataset, and adding more training samples from the CrackForest dataset does not improve this.

Analyzing the models with various dropout ratios reveals the effect of modifying



(a) DeepCrack Dataset



(b) CrackForest Dataset

Figure 14. Uncertainty, F1 score and calibration error on the DeepCrack and CFD test sets for models trained on the two respective datasets across dropout ratios

this important hyperparameter. Increasing dropout was expected to make the model less prone to overfit but also increase the prediction variance - This expectation bore out across both the baseline MC Dropout model, and the B-BACN. A consistent pattern of lower uncertainty, better calibration, and higher predictive performance across dropout ratios is seen for the B-BACN model, as shown in Figure 14.

Next, the effect of loss weighting strategies on model performance is analyzed. Figure 16 demonstrates that using a simple linear superposition of losses results in a higher epistemic entropy in most cases. The differences in aleatoric uncertainty and F-1 score are not significant. However, the sigmoid ramp-up technique produces

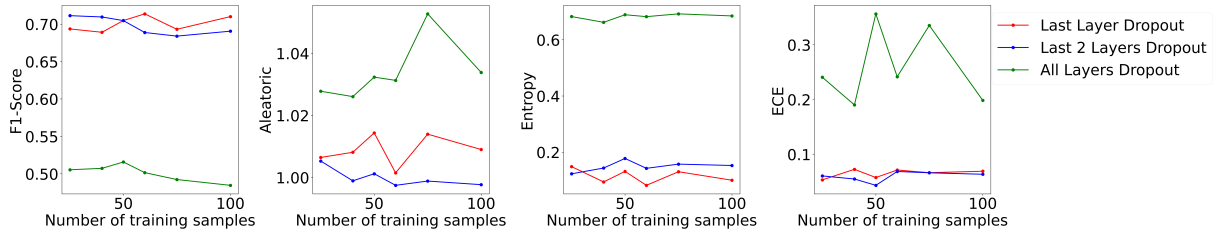


Figure 15. Effect of varying the number of dropout layers in the network on model performance and uncertainty.

marginally better predictive performance compared to all other techniques. The higher F-1 score when trained with 75 samples compared to 100 samples using the ramp-up technique is likely an artefact. Surprisingly, the model with the least F-1 score and the highest ECE was the CoV technique across all training samples.

The effect that the number of dropout layers has on model performance and uncertainty is analyzed next. Dropout is applied either on just the last layer, the last two layers, or all the layers of the network decoder. Generally, it is observed that the F-1 score of the predicted observations is highest when the number of dropout layers is limited to either just the final layer, or the last two layers. Due to the strong regularization effect offered by a dropout probability of 0.5 across all decoder layers, the F-1 score falls significantly. The aleatoric uncertainty is also higher for a network that is strongly regularized by dropout. It is found that the predictive entropy is significantly higher on the strongly regularized model, and the model is also less calibrated than the weakly regularized models in the other two cases.

One of the main objectives of predictive models is to ensure that a higher level of uncertainty is assigned to labels that deviate from the ground truth assignment, which can be verified in the validation set as the ground truth is available during the evaluation process. A demonstration of model calibration using the model trained



Figure 16. Model Performance Comparison on CFD Dataset with Varying Training Samples: Epistemic and Aleatoric Uncertainty, F1-Score, and ECE.

on the DeepCrack dataset is provided. Model calibration was first assessed on the DeepCrack test set and then the same model was evaluated on the CrackForest test set. which contains out-of-distribution examples. Model robustness was tested by introducing Gaussian noise with variance uniformly sampled between 0 and 1 to both test sets to simulate a challenging scenario. Figure 17 illustrates how the class-wise model uncertainty score aligns with the prediction quality, which is quantified by the class-wise F1 score. The points should be close to the line of equality (LOE) as shown in green, or demonstrate a negative correlation to uncertainty, which indicates that there is increasing uncertainty for samples that have a poor detection score in the test set. The first row of Figure 17 demonstrates that the DeepCrack background data is very well calibrated, but the crack class is less calibrated, with far more scatter across the predictive uncertainty. However, the calibration performance worsens with the out-of-distribution data in row 2. In row 3, however, noise with a mean of 0 and a variance between 10 and 50 is added. Introducing such high levels of noise shows the strong effect that it has on the model calibration. Model calibration methods such as the one shown in this study can therefore be used along with the expected calibration error to obtain more detailed evaluations of the image quality. Note that the linear fit in this figure is highly sensitive, especially in the noise injected case for the DeepCrack dataset, and the cross-dataset examples. The goal behind using this

Table 6. Performance comparison across uncertainty quantification (UQ) methods for models trained on CrackForest and DeepCrack datasets

UQ Method	Dataset	F1	Epistemic	ECE
MCD (Kendall and Gal, 2017)	CFD	0.710104	0.000529	0.068983
CD (Gal <i>et al.</i> , 2017)	CFD	0.700254	0.000090	0.075523
BBB (Blundell <i>et al.</i> , 2015)	CFD	0.700200	<u>0.005829</u>	0.059555
MCD (Kendall and Gal, 2017)	DeepCrack	0.761592	0.001005	0.122964
CD (Gal <i>et al.</i> , 2017)	DeepCrack	0.811375	0.000041	0.099586
BBB (Blundell <i>et al.</i> , 2015)	DeepCrack	0.781900	<u>0.005478</u>	0.070398

figure was to illustrate the relationship between the model uncertainty and the F1 score as an indicator of calibration.

The results end with a comparison of the MC-Dropout (MCD) method against the Concrete Dropout (CD) and the Bayes By Backprop (BBB) methods, and also mention some limitations of the MCD method that was the focus of this study. Table 6 shows model performance, model uncertainty and calibration error for models trained and tested on the CFD and DeepCrack datasets. Model performance does not significantly change for the CFD dataset. However, in the DeepCrack dataset, concrete dropout shows significantly higher F-1 score and significantly lower epistemic uncertainty. While the BBB method showed the best ECE metric, it slightly underperformed on both tasks in terms of F-1 score while producing a wider variance compared to the dropout-based models (underlined in Table 6).

While the dropout-based methods outperformed the BBB method, it should be noted that MC-dropout does not provide a posterior that directly depends on the number of samples n (Verdoja and Kyrki, 2021), such that the posterior narrows to zero at the limit of large data. MC-Dropout does provide larger variances and Concrete Dropout somewhat fixes this by driving dropout ratio p to zero in the limit

of large data. In the experiments, dropout ratio was initialized at 0.5, which then converged at 0.1 for both the CFD and DeepCrack datasets at the largest training data size. An important observation is that the epistemic variance obtained does not strictly reduce towards zero with more data. This behavior is better replicated in toy datasets but does not scale well to high dimensional datasets like those used in image segmentation. This study adds to the evidence that MC Dropout is a method that can be implemented with ease and can be modified to provide tighter uncertainty estimates. MC Dropout can also be a useful technique to provide indications of dataset shift. However, it comes with its share of downsides, and future work should focus on UQ methods that can more faithfully replicate a stronger increase in uncertainty with lower training data and in regions far away from the training data regime in high dimensions.

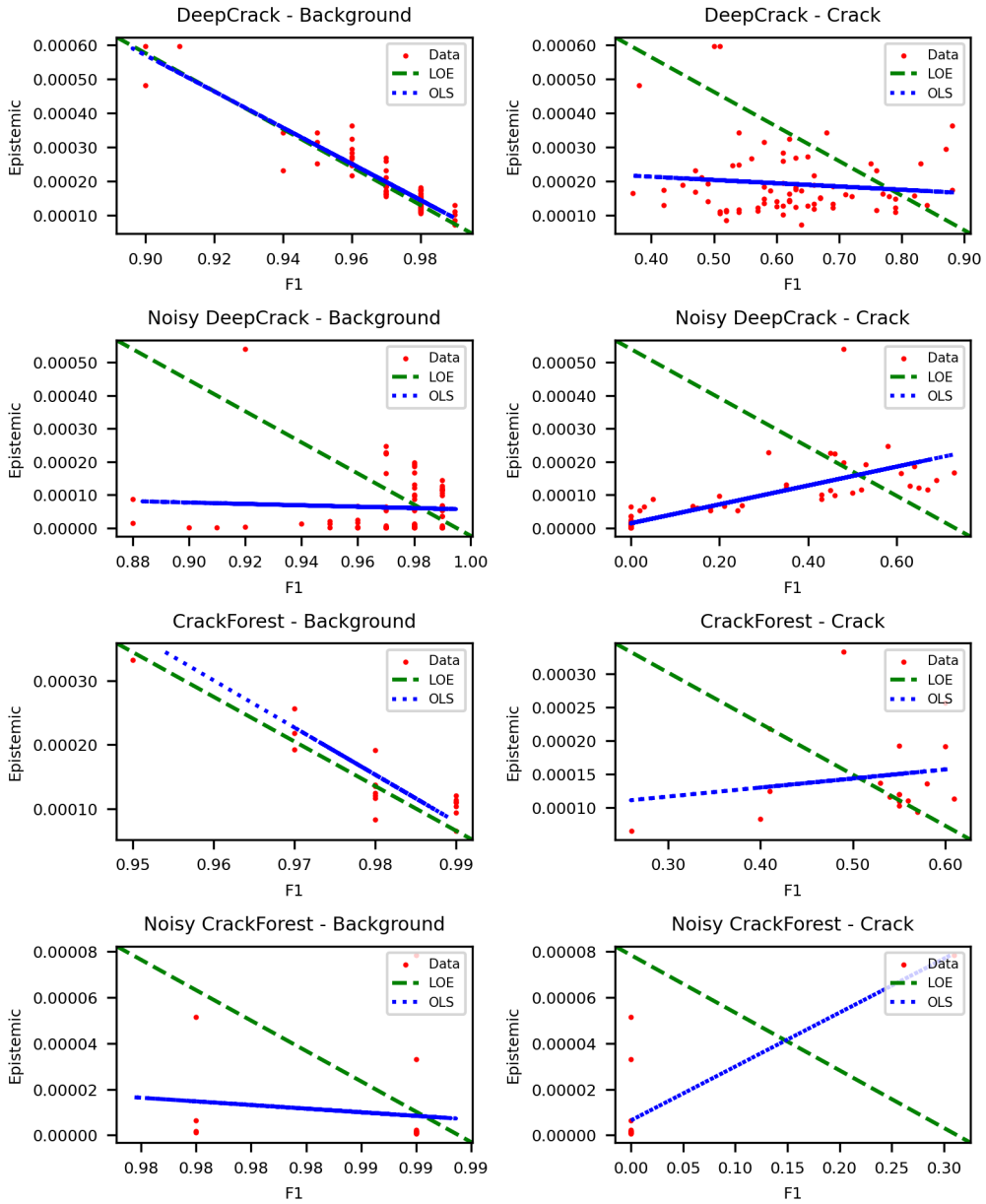


Figure 17. Epistemic Uncertainty and F1 score for a model trained on the DeepCrack dataset: (Row-1) Evaluation on the DeepCrack test set with No Noise, (Row-2) Evaluation on the DeepCrack Test Set with input gaussian noise $\sim U(0.0,1.0)$. (Row-3) Evaluation on the CrackForest test set with No Noise. (Row-4) Evaluation on the CrackForest test set with input noise $\sim U(0.0,1.0)$.

4.6 Conclusion

In this chapter, Bayesian-Boundary Aware Convolutional Network (B-BACN) are introduced, which incorporates uncertainty decomposition into an epistemic and aleatoric component. The approach shows significant performance improvements, while reducing uncertainty and improving the expected calibration error. The utilization of boundary loss functions in the B-BACN model not only refines pixel-level predictions but also contributes to reducing model size.

The analysis highlights important implications for reliability engineering, as it offers insights into the performance and calibration of crack detection models. By accurately quantifying uncertainty, the model aids in making more informed decisions regarding system reliability and maintenance.

Different Bayesian Deep Learning methods are compared, including MC-Dropout and Bayes By Backprop, and it is found that Concrete Dropout exhibited the lowest variance predictions and achieved superior performance on the DeepCrack dataset. These findings suggest that leveraging advanced Bayesian Deep Learning techniques can enhance the reliability of crack detection models. Furthermore, the investigation into distribution shift revealed that as test data deviated from the training distribution, model uncertainty increased and calibration decreased, highlighting the importance of accounting for distribution shifts in monitoring systems.

MC-Dropout provides lower variance predictions compared to Bayes By Backprop, but the lowest variance prediction was obtained by Concrete Dropout, which also achieved higher performance when evaluated on the DeepCrack dataset. Loss weighting strategies have also been shown to be key for optimizing learning, with the sigmoid ramp-up strategy for boundary losses providing much lower prediction uncertainties

and marginally better F-1 scores than a simple linear superposition of losses. More involved loss weighting schemes such as Coefficient of Variance weighting did not show significant improvements to detection performance or model calibration.

GEOMETRY-AWARE BAYESIAN NEURAL NETWORK FOR VISION-BASED
IN-LINE INSPECTION OF GAS PIPELINES

5.1 Introduction

The efficient operation of pipeline infrastructure is often hindered by risk factors such as cracks, corrosion pits, and welding defects. Various in-line inspection techniques, including Magnetic Flux Leakage (MFL), Ultrasonic Testing (UT), Electro-Magnetic Acoustic Transducer (EMAT), and Eddy-current Testing (ET) (Helifa *et al.*, 2006; Iyer *et al.*, 2012; Nakamura *et al.*, 2016; Sohn *et al.*, 2014), are commonly used for pipeline anomaly detection. However, these techniques are often expensive and bulky and may require pipeline operations to be interrupted. As an alternative, visual inspection offers a cost-effective and miniaturized solution to detect damage in pipelines. Vision-based inspection tools can be performed online without shutting down the system, depending on the robot hardware and pipeline internal operating pressure. Nevertheless, vision-based systems have their limitations, such as the inability to detect subsurface defects and lower reliability in assessing severity and measuring hairline-width defects. Therefore, it is beneficial to combine vision-based systems with other complementary sensing tools.

Recent trends in industrial inspection emphasize the use of multiple modalities of data to enhance the information provided by a single sensor (Liu *et al.*, 2015; Kong *et al.*, 2020). While multiple modalities of data have been used for pipeline inspection in the past, this study analyzes the use of commercially available depth

sensors to augment the RGB data provided by an optical camera. A lot of work has been done in the development of vision-based detection algorithms over the years for Structural Health Monitoring (SHM) (Hawari *et al.*, 2018; Xu *et al.*, 2021; He *et al.*, 2019; Shafeek *et al.*, 2004), but comparatively fewer studies have been conducted on stereo-vision-based inspection (Frank *et al.*, 2017; Huynh *et al.*, 2016, 2015; Alzuhiri *et al.*, 2021). Stereo-vision-based detection under uncertainty has not seen much attention either. The lack of depth information in evaluations with no stereo cameras produces several limitations, as there are no straightforward methods for producing any quantitative evaluations of surface defects, such as measurements of the defect area and depth. The setup designed by Alzuhiri *et al.* (2021) uses an assembly of a camera and a light pattern projector source for active stereo vision, known as structured light. The design of the inspection system used by Alzuhiri *et al.* (2021) also uses a camera system that sequentially captures images along the length of the pipe. This work, in contrast, seeks to study the extent to which a self-contained, commercially available active stereo-sensor can be used to detect and quantify defects and highlight its limitations and potential for use in the industry. This work also differs from Alzuhiri *et al.* (2021) in that images are captured by rotating the camera about the axis of the pipe, with it being pointed normal to its surface. This provides significant advantages in obtaining depth profiles of the defects.

In this chapter, a commercial stereo-sensor is evaluated for its suitability as a vision-based in-line inspection device, with a discussion on its viability for rapid inspection of gas pipelines. In addition, a feature fusion module that combines depth and RGB features and leverages the geometric features of the pipeline to extract more powerful representations is proposed. Additionally, this work applies the MC-Dropout technique used in Chapter 4 to decompose uncertainty for defect detection. The

results show that the proposed uncertainty-aware fusion network is able to provide better performance compared to using only RGB data in the pipeline dataset.

5.2 Background

5.2.1 Working Principle of Stereovision

Stereo vision relies on matching corresponding points in one camera to another camera. This requires camera calibration, followed by triangulation. This section contains a brief description of this process, but existing literature in photogrammetry provides more details Hartley and Zisserman (2003).

It can be inferred from Figure 18 that a point in an image can be projected along a ray to a point in the 3D space, and vice versa. The point determines the ray equation on the image plane and the camera's optical center. The projection of a pixel on the image plane onto the 3D space is only determined by the equation of the ray projected from the camera. In other words, the position of the 3D point subject to a scaling parameter for the depth is the only inference that can be made. On the other hand, it is possible to uniquely determine the position of a projected pixel from the 3D space, provided the location and orientation of the camera's image plane is known. In Figure 18, there are two calibrated cameras. In this context, calibration is the process of determining the camera's internal parameters and the camera's position and orientation. The calibration is represented with calibration matrices. If this matrix is known, it is possible to project a 3D point in the scene to a pixel on the camera,

provided the camera's field of view is high enough.

$$x = PX \tag{5.1a}$$

$$P = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_1 & r_2 & r_3 & t_1 \\ r_4 & r_5 & r_6 & t_2 \\ r_7 & r_8 & r_9 & t_3 \end{bmatrix} \tag{5.1b}$$

In Equation 5.1a and 5.1b, the pixel coordinate x is obtained by linearly transforming the world coordinates X using the calibration matrix P . The calibration matrix consists of the intrinsic parameters represented by f , p_x and p_y , and the relative rotation matrix and translation vector from a common coordinate system given by the r_i and t_i respectively. The inverse problem involves computing a point in the world corresponding to a point in the image. Given two cameras with known calibration matrices, an estimate without scale ambiguity can be made using triangulation. A point in the image plane of one camera can lie anywhere on a ray projected from the image plane. The intersection of the ray from the second camera with the ray from the first camera provides the corresponding match. The 3D point and the rays to the 3D point from each camera plane constitute the epipolar plane. The epipolar plane is constrained to be hinged to the line joining the optical centers of the two cameras, known as the baseline. In general, the correspondences of a point in the first image plane lie on a line called the epipolar line on the second image plane. This epipolar line is used for the matching point search. Determining the equation of this epipolar line requires a transformation from a point in the first image plane as:

$$E = t \times R \tag{5.2}$$

$$l' = Ex, \quad \text{such that } x'^T l' = 0$$

In Equation 5.2, t and R are relative translation vector and rotation matrix respectively for each image plane to obtain the essential matrix E . The epipolar line

l' on the second image plane is then computed from the essential matrix and a point on the first image plane denoted by x .

The search space for the matching process is made smaller by rectification - projecting the images from both cameras onto a common image plane. This puts the epipoles of both image planes at infinity. All epipolar lines are, therefore, parallel. The consequence of this projection is that the second image is essentially shifted along a line relative to the first image, removing the 3 relative rotational degrees of freedom and 2 translational degrees of freedom. Figure 18 (Right) shows the top-down view of the rectified configuration of the two-camera system. Using similar triangles, one can calculate the depth of a point, as shown below in Equation 5.3:

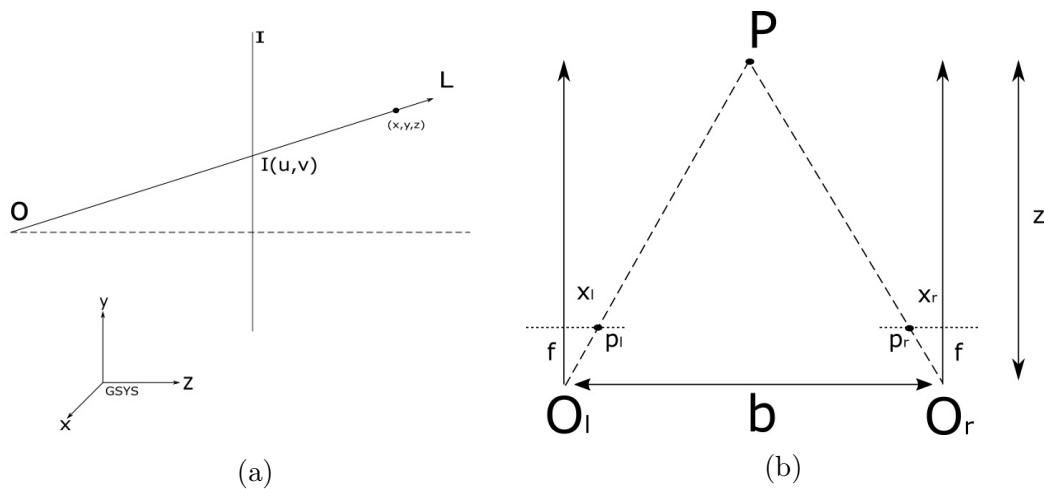


Figure 18. (a) Projection from the image plane to 3D world coordinates is only known upto scale (b) Triangulation approach to find the corresponding point on the second camera for a point on the first camera.

$$\frac{b + x_l - x_r}{z - f} = \frac{b}{z} \quad (5.3)$$

$$z = \frac{fb}{x_l - x_r} = \frac{fb}{d}$$

where b is the baseline, f is the focal length z is the depth of the point in the world frame, x_l and x_r are the horizontal coordinates in the image plane, and d is the disparity.

5.3 Methodology

5.3.1 Depth Sensor Model

For industrial inspection, the ideal outcome is to be able to detect defects, measure them, and quantify their respective uncertainties. This is a multi-step process, with uncertainties propagating from the sensor, into the detection model and finally manifesting themselves in the measurement. Any sensor output can be expressed in the form of a mean signal and a set of variances:

$$z = \bar{z} + \epsilon_s + \epsilon_t \tag{5.4}$$

where z is the signal, decomposed into its mean \bar{z} and variance ϵ . The variance consists of a spatial component ϵ_s and a temporal component ϵ_t . The overall uncertainty of the depth map is dependent on both the spatial and temporal noise of the sensor. The scope of this study is limited to characterizing the noise sources at the level of the signal output. Explicit analytical modeling of the causes of this noise is not performed. The noise can result from factors such as lens distortion and the resulting uncertainties in the intrinsic parameters. In addition, the pixel resolution limitation causes the quantization of disparity. The other source of noise is due to the matching process and the image quality. Generally, this is dependent on the application in consideration. The matching cost function for finding corresponding points can suffer

from ambiguities that lead to multiple matches with similar matching scores for in-line inspection. Filtering out the best match from a set of similar candidate matches has been done internally using the D435i stereo-matching algorithm.

To evaluate sensor performance, a normalized metric called the subpixel RMS error is utilized, which is obtained from fitting a plane onto a depth map of a flat wall. The plane-fit has an associated RMS error that provides an estimate of the spatial noise in the image, and the calibrated camera was found to have a temporally stabilized RMS error of 0.12%. The subpixel RMS error is given by $RMSE = \sqrt{z_i^2 - z_p^2}$ where z_p and z_i are the plane-fit projected depth and the actual depth, respectively. Due to the small error in the plane fit, there was no significant subpixel error. Knowing the RMS error of a plane fit only provides a metric for the inherent noise level in the sensor. However, this does not answer the crucial question of the minimum change in depth that the sensor can reliably detect, which is an important parameter that needs to be measured for optical metrology. For this examine the sensitivity of the depth function needs to be evaluated:

$$dz = \frac{-z^2}{fb}dd + \frac{z}{b}db + \frac{z}{f}df \quad (5.5)$$

Equation 5.5 is used to demonstrate the sensitivity of the depth to the parameters it depends on, which are the disparity d , baseline b and the focal length f . The focal length and baseline is fixed for the D435i, so the sensitivity of the depth to small changes in disparity is proportional to the square of the depth. This means that an infinitesimal change in disparity for an object that is close to the camera would cause a smaller change in depth compared to an object that is farther away. The consequence of this relationship is that finer depth sensitivities are possible for

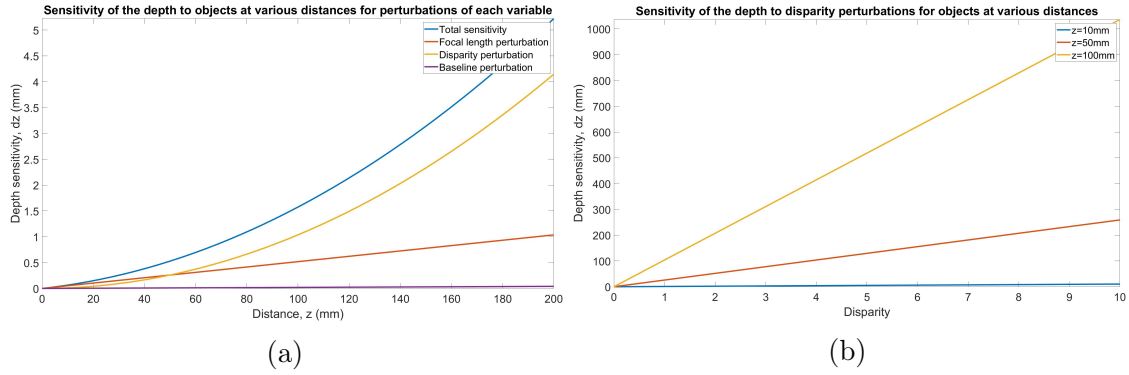


Figure 19. (a) Sensitivity of the depth map to regions at various distances in (mm) to perturbations in disparity, focal length, and baseline. (b) Sensitivity of the depth to disparity perturbations for objects at various distances

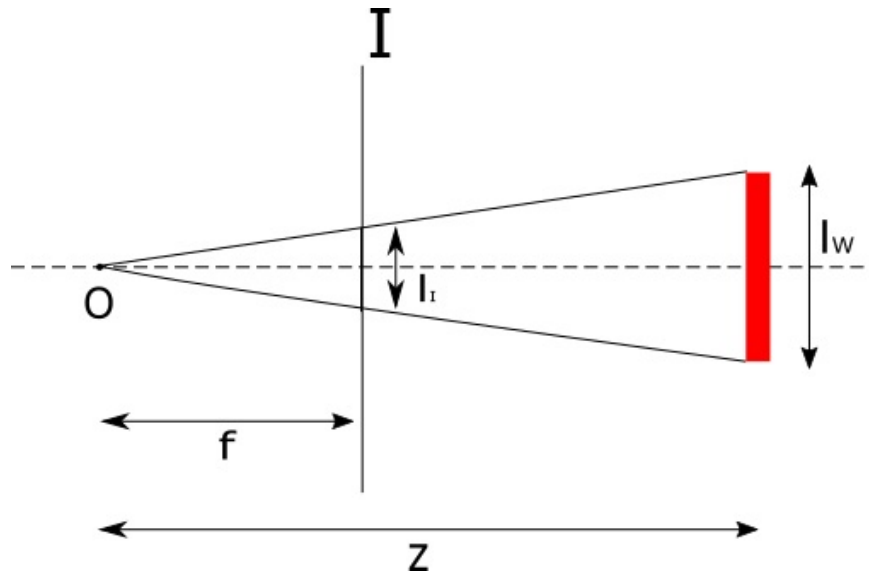


Figure 20. The relationship between the length of an object in image coordinates and the world coordinates can be determined using similar triangles

the same perturbation in disparity for objects that are closer to the camera. This is demonstrated in Figure 19.

$$\frac{l_I}{f_x} = \frac{l_W}{z} \quad (5.6a)$$

$$f_x = \frac{\frac{1}{2}X_{res}}{\tan(\frac{\theta_H}{2})} \quad (5.6b)$$

$$l_I = \frac{\frac{1}{2}X_{res}}{\tan(\frac{\theta_H}{2})} \frac{l_W}{z} \quad (5.6c)$$

In Equation 5.6, consider an object that is z mm away from the sensor. Assuming a pinhole camera model, the length in pixels l_I for an object of length l_W in the world coordinates can be derived using similar triangles as shown in Figure 20. To get the image length in pixels, the focal length also needs to be expressed in pixels, given by f_x . The camera field of view can be expressed as θ_H for horizontal, and θ_V for vertical. A similar relation can be derived for the height of the object and the height in pixels. The D435i has a pixel size of $3\mu m \times 3\mu m$. From Equation 5.6, the smallest object that the stereo-camera can detect can be found by computing $\frac{2zl_I}{f}$ with a length in image space of 1 pixel. For a defect that is 200mm away, which would be the case for the experimental setup that is used in this study for a 400 mm diameter pipe, the length captured by 1 pixel is 0.621mm.

5.3.2 Data Fusion for Defect Detection

In this section, the rationale behind using point-cloud derived features for defect detection and the processing of the point cloud is described. The purpose of using multiple modalities in prediction is to provide complementary information to aid model prediction. For industrial inspection, the RGB data stream provides texture and plane-projected morphological information without depth perception. The depth

stream provides a more fine-grained morphological representation of the scene. This study proposes a method to integrate data streams derived from the depth stream that would be relevant to the pipeline defect detection problem. Data fusion rules have been extensively studied in the field of image processing. Reviewing the literature reveals that the complexity of the fusion methods has increased to produce incremental improvements in quantitative performance metrics when tested on large datasets of natural images. However, concrete explanations as to why these modifications work and whether the improvements are predictably higher across different kinds of datasets are missing. The empiricism of deep learning architecture modifications led us to approach this part of the network with the aim of using a more transparent, explainable fusion rule architecture derived from classical decision-theoretic and data fusion principles. There are two principled ways of approaching data fusion in the context of deep learning: (i) Neural Architecture Search (Liu *et al.*, 2019a) (ii) Rules-based fusion (Gupta *et al.*, 2014). The first approach seeks to find a network architecture that can robustly fuse features from different data sources to satisfy a meta-objective of predicting well across multiple datasets. The second approach, while not directly geared towards improving detection performance metrics, would instead target providing interpretability for how the individual modality information is being utilized for the final prediction, and provide a way for the end-user to make operational choices for sensor integration. Industrial inspection in its current state could benefit from a hybrid approach of data-driven feature extraction, engineered data pre-processing and feature fusion rules, especially given that the number of samples available for training is not very high.

Fusion can be accomplished in multiple ways:

- Data Fusion: In this case, the fusion occurs at the level of the raw data.

- Feature Fusion: Features extracted from the data are fused together in intelligent ways to produce a new, combined feature representation.
- Decision Fusion: Several classifiers, each operating on one type of data are used to fuse the decisions made by each classifier to produce a combined evaluation using all the sources available.

In this work, new features are derived from the point cloud that were hypothesized to be relevant for pipeline data. The fusion rules for combining the RGB and point cloud-based information are then derived.

5.3.2.1 Feature Descriptor for Defect Detection: Depth Normal Curvature Representation

The problem of transforming the depth map of a cylinder to an equivalent flat plate with highlighted defect areas was studied by Alzuhiri *et al.* (2021). Alzuhiri *et al.* (2021) used a least-squares fit of an ellipsoidal surface to accomplish this task. But this approach is computationally intensive as it requires parametric modeling of pipeline geometry, and is sensitive to the location of the camera. Moreover, it was primarily used for the reconstruction of gas pipeline geometry. The objective here is to design feature descriptors from the point cloud that can potentially be extended for use in other domains such as road crack detection.

The novel DNC (Depth-Normal-Curvature) representation is now described. This is a 6-channel representation compressed into 3 channels using 1x1 convolutions, as shown below in Figure 21. This compression module is the first step in learning to weigh the engineered features to appropriately include the right proportions of the feature set from the point cloud data. The 1x1 is used for reducing the channel

dimension, and can take an input of dimension $N \times F_I \times H \times W$. If the kernel has F_O filters, then the output dimension is expected to be $N \times F_O \times H \times W$. The surface normal is a 3-channel image, one for each component of the normal vector. The curvature consists of the mean and Gaussian curvatures, which can be computed from the principal curvatures of a point on a surface. The depth information obtained from the stereo camera only showed weak signals of the defect. In contrast, the curvature and normal maps can highlight these features with strong signals, as shown in the demonstrative example in Figure 23. The point cloud has all of this information, and the goal is to find out whether transforming this point cloud data into the depth map, normal map and curvature maps would improve defect localization performance.

Another justification for using this approach is that the majority of the ILI procedure occurs in cylindrical sections. Note that the camera is placed such that it rotates about the axis of the pipe. This makes it possible to take advantage of the symmetry in the geometry of the view being captured. The setup also helps take advantage of the fact that the curvature of the cylinder is constant, except when there are imperfections on the surface, be they dents or pits. Another source that can cause a deviation from this ideal situation would be the noise inherent in the depth signal and the errors induced by the algorithm used to compute curvature. Similarly, for general detection and scene understanding problems, the surface normal provides additional geometric cues similar to the curvature. The normal serves to provide a "bump map", similar in vein to the ones used in computer graphics, to simulate wrinkles and bumps in textures. The differential geometry of a surface provides a discretizable formulation of normals from first derivative information and curvatures from second derivative information, assuming that these derivatives exist throughout the domain.

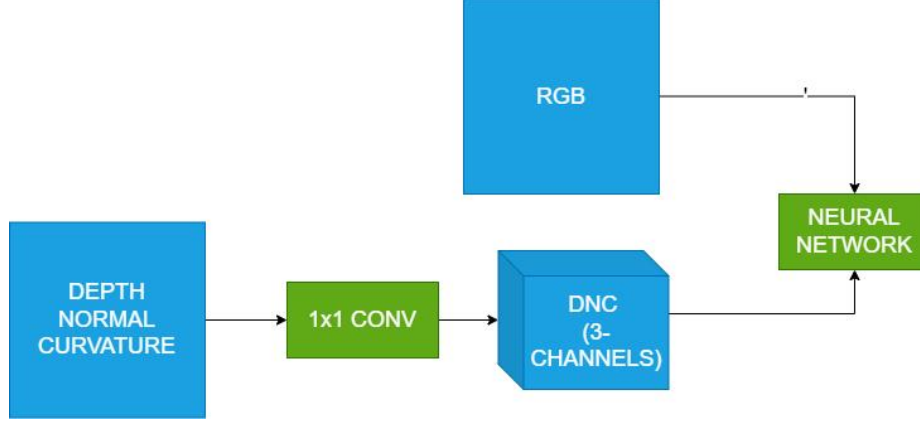


Figure 21. RGB-DNC Data Pre-processing Module: This module compresses the channel-wise dimension of the DNC input from 6 to 3 channels.

The surface normals and curvatures are derived from the depth map $Z = h(X, Y)$, which is essentially a Monge patch $\zeta : U \rightarrow R^3$ with the form:

$$\zeta(X, Y) = (X, Y, h(X, Y)), (X, Y) \subset U \quad (5.7)$$

The normals η can be calculated using the first derivative information and the curvature can be calculated using the second derivative information. The Gaussian and mean curvatures K and H of the surface are then computed using the standard equations describing the fundamental coefficients of a surface (Chase, 2015). The derivatives are computed along the x and y directions as follows:

$$D_i \zeta = \left[\frac{\partial X_j}{\partial x_i} \frac{\partial Y_j}{\partial x_i} \frac{\partial Z_j}{\partial x_i} \right] \quad (5.8a)$$

$$D_{ii} \zeta = \left[\frac{\partial^2 X_j}{\partial x_i^2} \frac{\partial^2 Y_j}{\partial x_i^2} \frac{\partial^2 Z_j}{\partial x_i^2} \right] \quad (5.8b)$$

$$D_{ij} \zeta = \left[\frac{\partial^2 X_j}{\partial x_i \partial x_j} \frac{\partial^2 Y_j}{\partial x_i \partial x_j} \frac{\partial^2 Z_j}{\partial x_i \partial x_j} \right] \quad (5.8c)$$

$$\eta(X, Y) = \frac{D_x \zeta \times D_y \zeta}{|D_x \zeta \times D_y \zeta|} \quad (5.8d)$$

The derivatives computed above in Equations (5.8a) - (5.8c) are used to compute the normal η in Equation (5.8d). The curvature information is derived from the fundamental coefficients of the surface by first computing the inner product between the first derivatives for the first 3 components E, F, and G. The inner product between the second derivative components and the normal map can be used for computing L, M and N:

$$E = D_x\zeta \cdot D_x\zeta; F = D_x\zeta \cdot D_y\zeta; G = D_y\zeta \cdot D_y\zeta; \quad (5.9a)$$

$$L = D_{xx}\zeta \cdot \eta(X, Y); M = D_{xy}\zeta \cdot \eta(X, Y); N = D_{yy}\zeta \cdot \eta(X, Y); \quad (5.9b)$$

$$K = \frac{LN - M^2}{EG - F^2} \quad (5.9c)$$

$$H = \frac{LG + NE - 2FM}{2(EG - F^2)} \quad (5.9d)$$

Equation (5.8d) and Equations (5.9a) - (5.9d) is used to obtain representations of simple geometric shapes.

Using the above formulation, a set of simulated surfaces are now used for analyzing the features. Consider a discrete smooth flat plate that was created with a (1000x1000) grid with $z = 1$ as shown in Figure 22. The normals were calculated to be (0,0,1) across the entire grid, which is accurate. Both mean and Gaussian curvatures are zero, which tells us that the computation is correct for the flat plate. Next, a cylinder section was created using a (1000x1000) grid, with a 180-degree field of view as shown in Figure 22. The mean curvature was calculated to be -1, and the Gaussian curvature was calculated to be 0. The cylinder that has been created now can be thought of as bending the flat plate about the longitudinal axis. While the surface shape has changed, the length of the arc connecting two points in that surface has not. The Gaussian curvature formulation outlined above is invariant to such transformations

and remains unchanged, but the mean curvature is not invariant to the change in the embedding of the surface within the 3D space. The next case that is considered is the rotated cylinder. This case demonstrates a scenario that can occur during data collection from the pipe sample. Due to the misalignment of the camera, it is possible that the depth map can deviate from being approximately normal to the surface. The curvatures demonstrate that they are approximately invariant to such rotations (Besl and Jain, 1986).

5.3.3 Fully Convolutional Neural Networks for RGB-D Segmentation

The baseline architecture is based on the fully-convolutional encoder-decoder network (Long *et al.*, 2017) as shown in Figure 24. The encoder has two parallel streams, one for 3D data and the other for RGB data, and the output of each of the convolutional sub-blocks in the encoder is extracted for feature fusion. This provides a hierarchy of features at multiple scales, with earlier layers extracting fine-grained morphological information, and later layers extracting coarse-grained category and location information. The resulting layers are then passed into the decoder, which consists of transposed convolutions, to upsample the compressed representation into the label space. The feature fusion modules are derived to prioritize the resulting feature maps from the encoder convolutional blocks so as to lower the training loss. The feature fusion schemes are visualized below the main architecture diagram in Figure 24. The base case for the fusion block is the element-wise sum, abbreviated in the results as (SF). The details pertaining to the more sophisticated fusion blocks derived are described below:

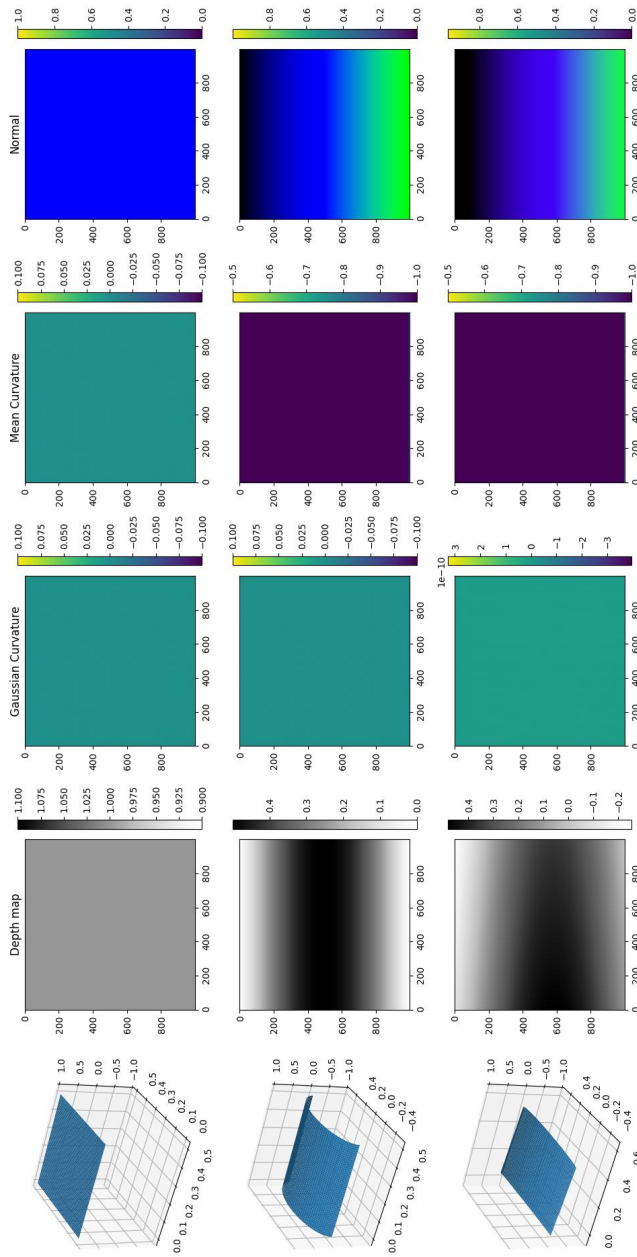


Figure 22. Feature descriptors for various rigid objects demonstrating the various invariances that curvature possesses - By row from top to bottom: Flat Plate, Smooth Cylinder, Rotated Smooth Cylinder - Euler Angles $[15^\circ, 15^\circ, 0^\circ]$

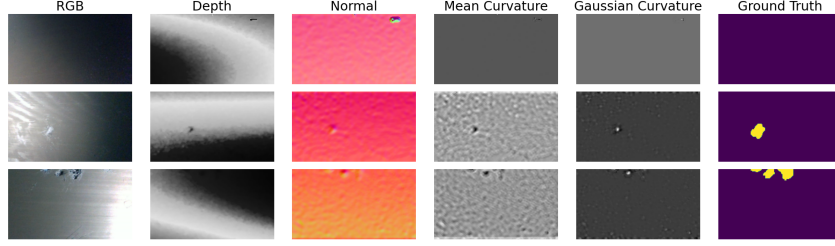


Figure 23. Comparing the RGB-D and RGB-DNC data input formats: RGB-DNC provides a more refined representation of defects at the input level.

5.3.3.1 Weighted Non-Linear Fusion (WNLF)

The first approach implements a method to fuse data at the feature level. A non-linear weighted combination of the max pooling layer outputs from each convolutional block is performed as follows:

$$F = f(M_{RGB}, M_D) \quad (5.10a)$$

$$f(M_{RGB}, M_D) = \text{ReLU}([M_{RGB}, M_D] * K) \quad (5.10b)$$

In Equation 5.10a, M_{RGB} and M_D are the max-pooling output after each convolutional block of the encoder. The max-pooled outputs are first concatenated, then convolved with a 1x1 convolution kernel K and finally passed through a ReLU non-linearity, as seen in Equation 5.10b. Concatenation of the RGB and depth channels doubles the channel dimension length. The convolution acts on the two sections of the concatenated representation to reduce the channel dimension back to its original length. For example, the first block of the encoder leaves us with a max-pooled output representation that has 8 channel dimensions each for the RGB and depth data. Concatenating these features along the channel dimension gives us a total of $C_I = 16$ channels. The fused representation obtained from Equation 5.10b returns back the original 9 dimensions. Let the input features have a dimension of $C_I \times M \times N$. The 1x1 convolution layer has a

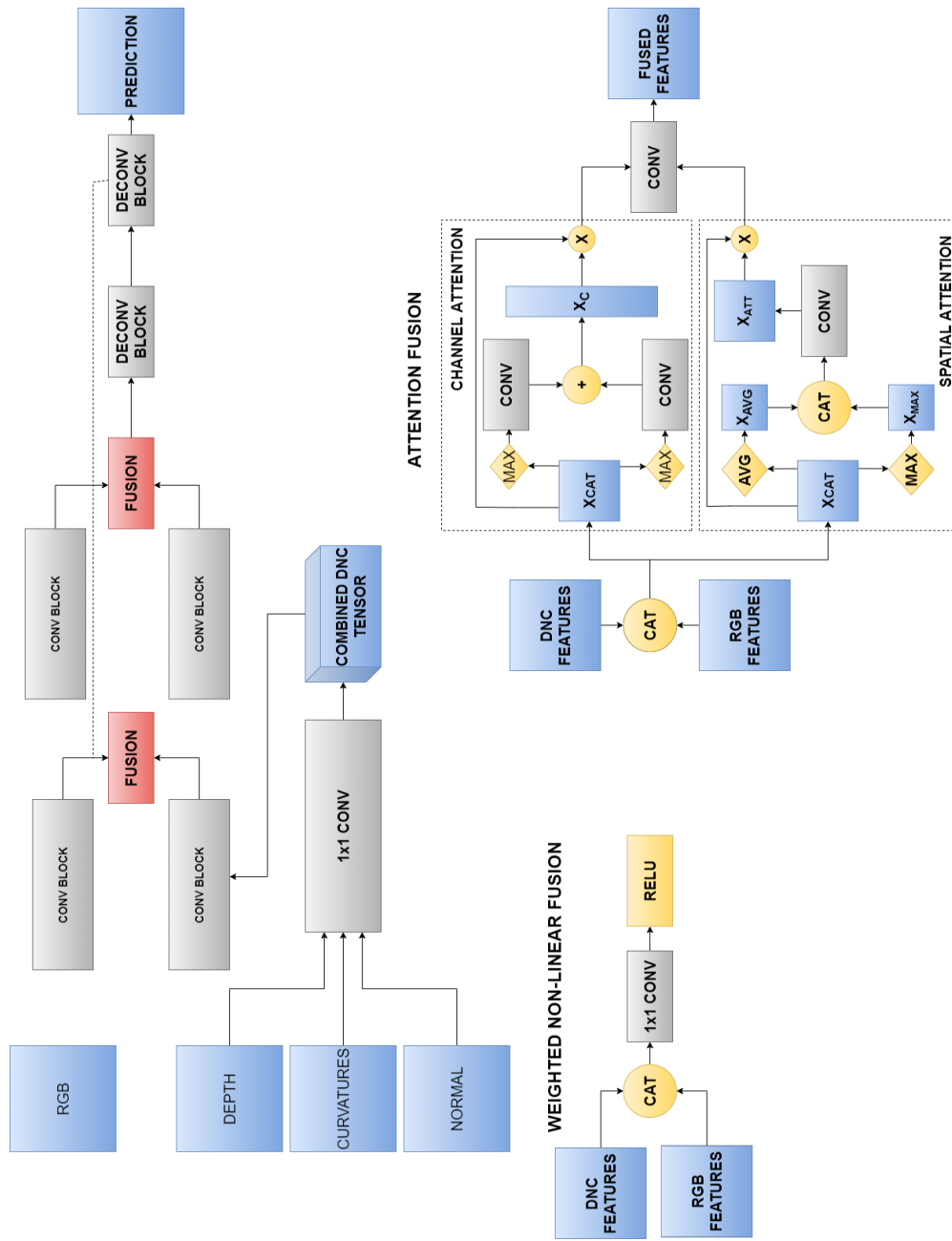


Figure 24. Overview of the proposed network architecture: Overall network architecture diagram for feature fusion in semantic segmentation. Two fusion schemes proposed in this chapter are showcased here.

shape $C_O \times C_I \times 1 \times 1$. Each kernel K_i produces a distribution of weights, when convolved with the input features produces a weighted sum $\sum_{j=1}^{C_I} K_{ij} \cdot I_j = F_i$. Each convolved output F_i is one component out of the total C_O components at the output of the fusion layer. From the point of view of interpretability, this is not the same as a simple weighted additive combination. Instead, an output map that consists of a set of weighted combinations is obtained, densely weighted by each convolution kernel, shown in row representation as:

$$\begin{aligned}
 K_1 &= [K_{11}, K_{12}, \dots, K_{1C_I}], \\
 K_2 &= [K_{21}, K_{22}, \dots, K_{2C_I}], \\
 &\vdots \\
 K_{C_O} &= [K_{C_O1}, K_{C_O2}, \dots, K_{C_OC_I}]
 \end{aligned} \tag{5.11}$$

5.3.3.2 Attention Fusion Block (AF)

The second approach, the Attention Fusion Block, is designed to combine RGB and 3D data streams by performing more sophisticated operations on the intermediate features to weight them by important channels and spatial regions. To do this, it employs two attention mechanisms, channel and spatial attention. This structure was proposed in (Woo *et al.*, 2018) for RGB object detection, and has been used for various image classification and detection tasks. It is natural to extend its use for semantic segmentation with multiple data sources.

$$F_{attention} = g(M_{RGB}, M_D) \tag{5.12a}$$

$$g(M_{RGB}, M_D) = S(C(M_{RGB}, M_D)) \tag{5.12b}$$

In Equation 5.12a, M_{RGB} and M_D represent the input feature maps from the RGB and depth data streams, respectively. Equation 5.12b elaborates the process, where $C(\cdot)$ represents channel attention and $S(\cdot)$ represents spatial attention.

- **Channel Attention (C):** The channel attention mechanism focuses on the channel-wise feature weighting task. It takes the feature maps M_{RGB} and M_D and applies an adaptive average pooling and an adaptive max pooling so as to reduce the spatial dimension to (1,1), followed by two convolution layers - One that reduces the channel dimension, and followed by the second that brings it back to the original dimension. Both the maxpooled and average pooled representations are then combined via addition to return a set of channel-wise attention weights. These weights are applied to the original feature maps to emphasize more relevant channels.

$$X_{Fused} = Cat(M_{RGB}, M_{3D}) \quad (5.13a)$$

$$C(M_{RGB}, M_{3D}) = X_{Fused} \odot W_{Fused}^c \quad (5.13b)$$

where \odot denotes element-wise multiplication, and W_{Fused} is the learned attention weights for RGB and 3D channels.

- **Spatial Attention (S):** After the channel attention step, spatial attention is applied, focusing on "where" to emphasize in the spatial domain. In contrast with the channel attention module, the spatial attention module utilizes both average and max pooling operations across the channel axis to generate spatial attention maps. These maps are then convolved to assign spatially varying importance to different regions of the feature map.

$$S(x) = \sigma(Conv(cat([AvgPool(x), MaxPool(x)]))) \quad (5.14)$$

where cat denotes concatenation, $Conv$ a convolution operation, and σ the sigmoid function for normalization.

The resulting output $F_{attention}$ from the Attention Fusion Block is a set of feature maps where both channel and spatial aspects have been weighted, emphasizing the most informative features for subsequent decoding in the segmentation task.

5.3.4 Post-processing and Defect Measurement

Once the segmentation is complete, the one-hot encoded prediction and the point cloud maps are used to evaluate the sizes of the defects. The overall process is shown in Figure 25. While the prediction can be used directly, the point cloud maps need some post-processing before its usage for measurement. It is possible that the depth map contains holes due to the block-matching algorithm. These holes are known as invalid depth pixels. To remove these invalid pixels, interpolation is performed between values of depths. If the invalid regions are small compared to the area of the entire depth map, then the interpolation does not severely corrupt the geometry of the point-cloud, but assumes a smooth, continuous transition.

One-hot encoding used to separately evaluate the corrosion and crack defect. The detected contours are then sorted for both defects. The list of extracted contours are then used to compute defect characteristics such as area, depth, width and length. For cracks, all four characteristics are reported, and for corrosion, it is sufficient to report only are width and length.

Contour area is computed by calculating the convex hull of the contour. Contours are filtered by checking if a defect occupies less than 4 pixels in the image. Depth computation is done using a bounding polygon of the defect. For this study, a

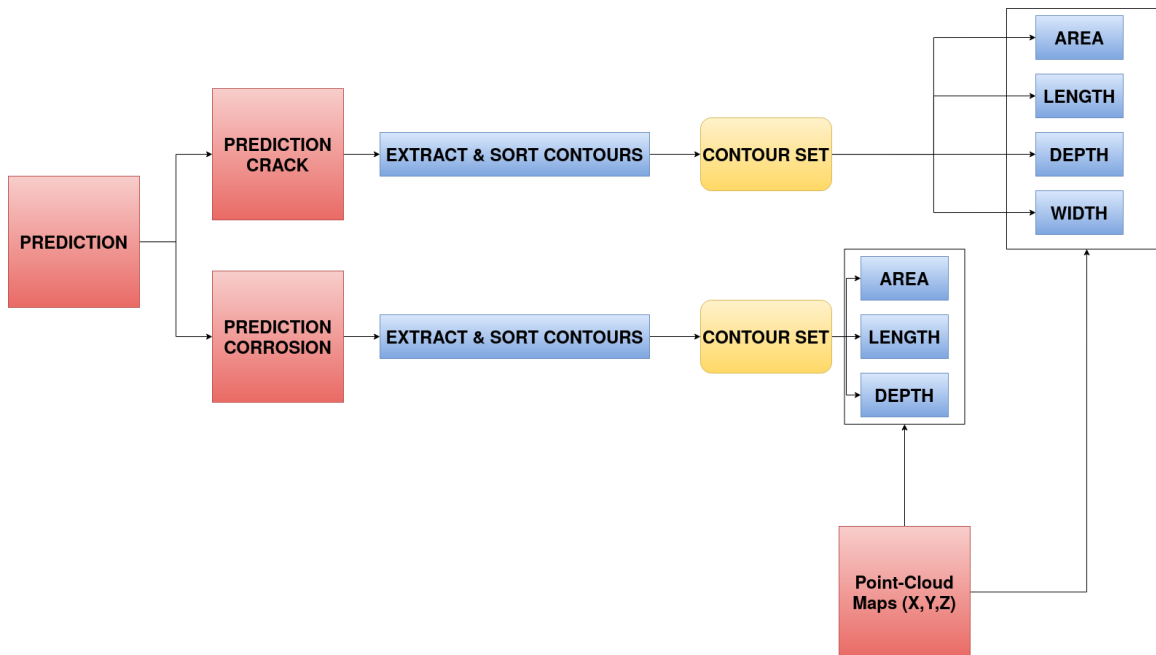


Figure 25. Post-Processing of the segmentation and point-cloud maps to extract crack and corrosion defect measurements using contour analysis.

rectangular approximation is used for the polygon. If the defect is small, the effects of curvature on the depth is minimal. The maximum depth of the defect is computed as the difference between the bounding rectangle regions' maximum and minimum depth. Contour length and width are computed by using the minimum area rectangle surrounding the contour. The width is computed by calculating the euclidean distance between the left and right side of the bounding rectangle, and the length is computed as the diagonal length of the rectangle. These metrics for defect quantification are calculated using image processing functions from OpenCV (Bradski, 2000).

5.4 Experimental Setup

5.4.1 Dataset

Table 7. D435i settings for image acquisition.

Parameters	Baseline values
RGB Resolution and Rate	1280x720 @ 30 Hz
RGB Field of View	69°x42°
Depth Resolution and Rate	1280x720 @ 30 Hz
Depth Field of View	87°x58°
Focal Length	1.93 mm
Baseline	50 mm

The platform used for collecting data uses off-the-shelf hardware. These include the sensors, the computing system, and the robot platform. The primary purpose of this experimental setup was to acquire data for the main dataset that this chapter discusses, the RGB-D pipeline defect dataset. The platform consists of an active stereo camera, the Intel RealSense D435i. The baseline parameters for the Intel RealSense D435i depth camera are shown in Table 7.

The ASU pipeline RGB-D data consists of 77 images. To my knowledge, there has been no RGB-D dataset for pipeline inspection published for benchmarking till date, so this dataset provides corrosion pitting and crack defects for segmentation tasks. To increase sample diversity, the data is augmented using random flips during training. Operations such as rotation and shear unrealistically distorts the depth, normal and curvature information, so these transformations were not used. The dataset has 2 defect categories: pitting defects and cracking defects. The ASU pipeline dataset was primarily used to evaluate the comparative performance of feature fusion with various types of input data with the RGB image, across dropout ratios and across

training samples, for model performance and uncertainty. Due to the small size of the dataset, 5-fold cross validation is used to average the results. In order to minimize the influence of class imbalance in the detection performance, the weighted cross-entropy loss term using the median of the class frequencies in each image is employed

5.4.2 Implementation Details

The network architecture used is an Encoder-Decoder model similar to those seen in previous chapters, with a transposed convolutional decoder, as explained in Section 5.3.3, and the experiments are conducted using PyTorch. Since the dataset is small, this study uses 2 encoder blocks, and 1 decoder block. Each block consists of convolution followed by ReLU and max pooling. A small network is sufficient to capture the features for corrosion and cracks without overfitting on the training subsets during cross validation. Similar to the previous chapter, the SGD with momentum is again used for optimization, with a momentum value of 0.9 and weight decay of 10^{-5} , along with a step-decay learning rate scheduler. Details on the metrics are given in Section 4.4 of Chapter 4. Similarly, the ABL and IOU loss functions are also used to enhance defect boundary reproduction, as explained in Chapter 4.

5.5 Results and Discussion

Quantitative out-of-sample performance metrics for the ASU Dataset is shown in Tables 8 and 9. Table 8 shows a substantial improvement seen by adding depth and curvature data all cases, with the normal maps providing an additional boost. Using

depth alone does not provide the strong signal for classification that the additional modalities do. The sampling variance is comparable and small across the board.

Model	mF1	F1 (Pit)	F1 (Crack)	Variance $\times 10^{-3}$
FCN (RGB)	63.16	60.13	30.06	0.3
AF (RGB-DNC)	<u>68.97</u>	65.39	<u>42.18</u>	0.5
AF (RGB-DC)	67.17	63.65	38.55	0.5
AF (RGB-D)	60.11	58.82	22.24	0.5
WNLF (RGB-DNC)	70.34	65.03	45.62	0.8
WNLF (RGB-DC)	68.46	64.48	41.61	<u>0.4</u>
WNLF (RGB-D)	59.48	62.92	16.24	<u>0.4</u>
SF (RGB-DNC)	67.93	62.48	42.04	0.8
SF (RGB-DC)	67.79	63.16	40.88	<u>0.4</u>
SF (RGB-D)	61.15	61.50	22.72	0.6

Table 8. Model performance comparison for segmentation tasks

Dropout p	mF1	F1 (Pit)	F1 (Crack)	Variance $\times 10^{-3}$
0.1	68.97	65.39	42.18	0.5
0.2	70.53	64.23	48.03	1.4
0.3	69.78	65.97	43.97	1.5
0.4	67.90	60.39	44.06	2.1
0.5	64.84	59.98	35.32	5.8

Table 9. Performance of RGB-DNC using the AF Fusion Model with Different Dropout Probabilities

From the results, it is clear that the additional data provides a substantial improvement to performance but no real improvements to model uncertainties. The defect measurement results is shown in Table 10. The epistemic uncertainty in prediction is transferred to the defect measurement and is evaluated by computing the variance of the MC-sampled predictions. The table demonstrates two examples of defect measurements - The first is a combined pitting and cracking defect. The model does not detect the crack, but manages to detect the pitting defect. Even though there are

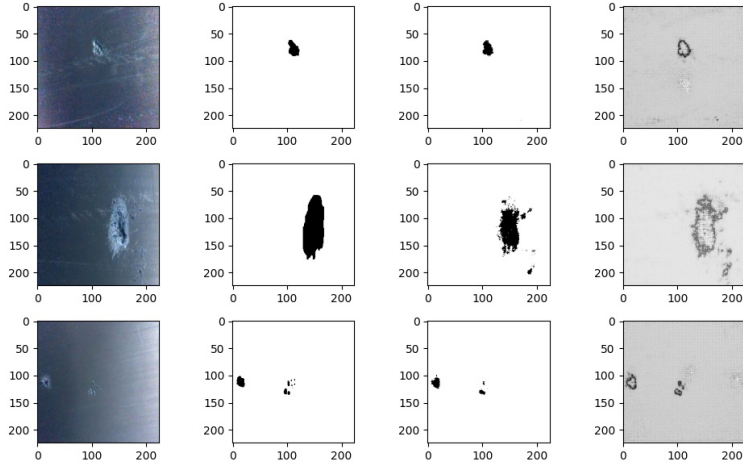
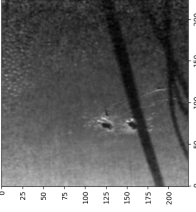
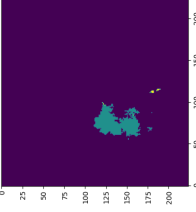
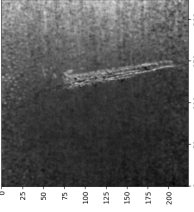
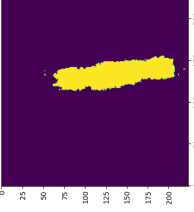


Figure 26. Demonstrative examples of detections in the ASU Pipe Dataset. Images from Left to Right: Input, Prediction, Ground Truth, Total Uncertainty

two separate pits, the combined area of both the pits were used to compare the mean prediction area with the area computed from the ground truth. The depth and length are also compared with the ground truth segmentation. The second image is a crack defect, which was detected and the estimated area, depth, length and width of the crack are compared. The depth sensor was found to be unreliable below the millimeter scale, with improved detections of defect depths in the millimeter scale. Signals of interest captured at sub-millimeter scales can be used for anomaly detection, but the corresponding depth measurements will not be reliable for measurement purposes. In the context of pipe damage modeling, this situation corresponds to early detection for corrosion and oxidization of metallic surfaces (Zhang and Ma, 2019). At the early stages of surface degradation, the dominant anomaly signal is increased metallic surface roughness, which can be detected but not precisely profiled along the depth direction by the stereo setup. Further improvements to the depth resolution for optical metrology at sub-millimeter scale using depth sensors is an important future research direction.

Table 10. Defect measurement demonstration from the ASU Pipe dataset

Image	Prediction	Defect	Measurement	From Prediction - Mean (SD)	From Ground Truth
		Pitting	Area (mm^2) Depth(mm) Length(mm)	1577.47(74.56) 4.93(0.09) 44.98(0.84)	1028.01 4.79 37.74
		Crack	Area(mm^2) Depth(mm) Length(mm) Width(mm)	- - - -	732.79 2.4 57.24 7.9
		Crack	Area(mm^2) Depth(mm) Length(mm) Width(mm)	3660.01(29.20) 2.11(4.23) 86.73(1.54) 23.02(0.28)	2353.41 0 79.92 17.05

The detections for the ASU pipe dataset is shown in Figure 26. The uncertainty maps show that the source of uncertainty is largely at the boundaries of the detected defect. The variance in defect area profiles is useful for downstream prognostics, and the ability to obtain uncertainty estimates along with defect morphology is essential for detailed risk evaluations. This method when applied to other sensor types such as higher resolution depth cameras, Time of Flight and Structured Light sensors can yield both depth and area profile variances with no changes to the neural network architecture, giving practitioners and operators the ability to use transfer learning to adapt their model to new types of sensors.

5.6 Conclusions

In this chapter, RGB and point-cloud features are fused for uncertainty-aware crack and corrosion detection in gas pipeline images. The utility of point-cloud derived 2D representations such as the Depth, Normal and Curvature representation are evaluated for uncertainty reduction and detection performance. The motivation behind using this representation was to leverage the geometry of the pipe effectively. To utilize the information from these modalities effectively, a weighted combination approach is proposed within the network encoder to fuse the data along multiple layers of the network. The findings show significant improvements to mean F1 (mF1) and class-wise F1 score when using curvature and normal maps. This is because the features take advantage of the geometry of the pipe. The depth resolution is found to be an important limitation of existing depth sensors, as it cannot produce precise depth evaluations beyond the millimeter scale owing to resolution limitations and noise. Active stereo is generally suited for rapid visual inspection of pipe infrastructure, and

incorporating detailed uncertainty maps along with detections and segmentations is recommended as a useful addition to existing damage evaluation methods.

DEFECT SEGMENTATION WITH LIMITED LABELED DATA USING
CONSISTENCY REGULARIZATION AND ACTIVATION MAP
INTERPOLATION

6.1 Introduction

Automated defect segmentation plays a crucial role in quality assessment for manufacturing processes. However, the acquisition of labeled data for defect segmentation is a formidable challenge. Labeling datasets is costly and often requires expert annotators (Rasmussen *et al.*, 2022). Moreover, defects in industrial settings pose unique difficulties, including intricate morphological characteristics, high intra-class variability, inter-class similarities (Chang *et al.*, 2012; Gyimah *et al.*, 2021), and the presence of minuscule defects (Ling *et al.*, 2022; Liu and He, 2022), which can be particularly elusive.

In this chapter, a semi-supervised method that leverages a small set of labeled examples and a larger set of unlabeled samples is used for defect detection. This approach is motivated by the need to alleviate the burden of data labeling, especially when dealing with fine-grained defect detection at the pixel level.

The novelty of this method is obtained based on three fundamental cornerstones: the smoothness assumption, the cluster assumption, and consistency regularization. The smoothness assumption posits that the decision boundaries between classes should be well-separated, while the cluster assumption suggests that classes should not overlap

but remain distinct. To enforce these assumptions, consistency regularization is used, which encourages the model to produce consistent outputs for perturbed inputs.

However, applying consistency regularization to image segmentation is not straightforward. Unlike image classification, segmentation requires pixel-wise predictions and is sensitive to input augmentations like rotation or translation. Note that the cluster assumption may not hold uniformly across image data (French *et al.*, 2019). To address these challenges, this method employs input interpolations in the low-dimensional embedding space to ensure consistency while utilizing common image augmentation techniques to enhance sampling diversity.

This chapter is organized as follows: Section 6.2 outlines key advances in image segmentation for defect detection and semi-supervised learning. Section 6.3 details the proposed semi-supervised method. Section 6.4 presents the training procedures and datasets. Following that, Section 6.5 presents evaluations of the proposed approach across the datasets considered, across multiple training settings. Finally, Section 6.6 concludes by summarizing the main results and outlines the implications and limitations of this study.

6.2 Related Work

6.2.1 Image Segmentation for Defect Detection

Early methods for defect detection focused on hand-engineered feature classification and defect classification using classical machine learning and statistical techniques (Liu *et al.*, 2019b; Suvdaa *et al.*, 2012; Liu *et al.*, 2016; Shi *et al.*, 2016). (Liu *et al.*, 2019b) used a multi-block local binary pattern (LBP) to filter features across multiple scales.

(Suvdaa *et al.*, 2012) uses SIFT features followed by an SVM for defect classification. (Liu *et al.*, 2016) uses image binarization optimization using genetic algorithms to segment defects in an unsupervised fashion, by using a thresholding approach based on inter and intra-class variance statistics. In most of the classical image processing-based approaches, the process of detection is multi-stage, requires lower amounts of training data and has simpler mathematical models, and complicated algorithmic models.

Classification based on deep learning has been explored by several works (Liu *et al.*, 2020b; Saiz *et al.*, 2018; Chen *et al.*, 2018). (Liu *et al.*, 2020b) uses a CNN architecture with fixed scales at two image resolutions, which compute features at the corresponding scales and are then concatenated for defect classification. (Chen *et al.*, 2018) uses an ensemble of three networks to classify defect images, and (Saiz *et al.*, 2018) combines pre-processing steps with a CNN in an attempt to increase the robustness of the CNN to inter-class defects. Semi-supervised classification algorithms such as (Di *et al.*, 2019; Wang *et al.*, 2021c) use varied methods to distinguish between textural samples with limited labeled data. (Di *et al.*, 2019) uses a large unlabeled corpus to train a convolutional auto-encoder and a GAN is then used to increase the sample size. Finally, the encoder of the trained auto-encoder model is used to extract features from the data, and is fed into a softmax layer for classification. (Wang *et al.*, 2021c) uses a graph-based deep network to perform classification. However, classification does not give us any indication of object location, without further post-processing. This gap was filled by various object detection (Li *et al.*, 2018, 2019; Zhang *et al.*, 2020) and semantic segmentation methods. The latter produces pixel-wise predictions, with various architectural modifications proposed recently to boost the performance in the validation and test sets, as in (Tabernik *et al.*, 2020; Dong *et al.*, 2020; Cheng and Yu, 2021; Yang *et al.*, 2020b; He and Liu, 2020).

Recent advancements in defect segmentation have focused on improvements and modifications to the deep feature extractors. For instance, the "MemSeg" model presented a memory-based segmentation network for real-time, high-accuracy semi-supervised surface defect detection (Yang *et al.*, 2022). MemSeg leverages a combination of artificially simulated abnormal samples and a set of normal samples to aid anomaly detection. This work sees the application of the oft used multi-scale feature fusion technique and a spatial attention module. Lightweight models are sought after in industry due to their prediction speed. In order to address this, a lightweight model for online surface defect segmentation on aluminum strip production lines was introduced, balancing segmentation speed and precision (Lv *et al.*, 2023). This work also utilizes modifications in the network architecture, such as spatial attention and feature pyramid networks (Lin *et al.*, 2017). Another approach is the MLR-Net, a multi-layer residual convolutional neural network designed for leather defect segmentation, which integrates spatial and channel attention mechanisms (Iqbal *et al.*, 2023). A sub-region U-Net model was proposed for segmenting defects, especially effective in scenarios with low contrast and high background noise (Zhu *et al.*, 2023). It is observed that the application of spatial and channel attention mechanisms across these models emphasizes the trend toward more precise and targeted feature extraction, which is crucial for accurate defect segmentation.

6.2.2 Semi-Supervised Segmentation

Labeled data acquisition has been a significant challenge for the defect segmentation task. While supervised methods offer the promise of precise pixel-wise predictions that can inform downstream decision-making, the task of labeling each pixel in an

image is a monumental effort that takes manpower and domain knowledge. This has led to efforts in multiple directions that aim to drive down the need for large amounts of labeled data. (Fernández-Moreno *et al.*, 2023) explored the trade-off between performance and annotation complexity in semantic segmentation, assessing various methods across different levels of supervision. The task of semi-supervised learning has been explored using the following approaches that regularize unlabeled outputs during training: Pseudo-Labeling (Lee, 2013; Chen *et al.*, 2021; Sime *et al.*, 2022; Xu *et al.*, 2023), consistency enforcement (French *et al.*, 2019; Yun *et al.*, 2019; Zhang *et al.*, 2017a; Verma *et al.*, 2019; Shi *et al.*, 2023; Hu *et al.*, 2022), adversarial training (Miyato *et al.*, 2019; Zhang *et al.*, 2017b) and co-training (Qiao *et al.*, 2018; Fan *et al.*, 2022).

The pseudo-labeling approach takes in a corpus of labeled and unlabeled samples, and generates fake targets for the unlabeled samples (Lee, 2013). More recently, a Cross-Pseudo Supervision (CPS) method has been proposed that takes the labels generated from the main network is used as supervision for unlabeled samples (Chen *et al.*, 2021). However, the problem with pseudo-labels is that the errors made in the target generation have no mechanism of correction. As a result, using incorrect pseudo-labels will tend to propagate these errors through the model during backpropagation, and cause a positive feedback loop of erroneous predictions leading to incorrect weight updates.

Methods to boost pseudo-labeling and correct for these accumulating errors have been proposed. Self-training improves upon the pseudo-labeling approach by successively adding new samples into the corpus and by selecting the samples that it is most confident about. (Sime *et al.*, 2022) proposed Semi-Supervised Pairwise Similarity Map Consistency and Ensemble-Based Cross-Pseudo Labels (SimCEPS) that uses an

ensemble of outputs from multiple predictor decoders. Regional Contrastive learning (ReCo) (Liu *et al.*, 2021) is a technique that applies a contrastive loss on the latent embedding space before the decoder, encouraging similar classes to cluster closer together, providing significant performance gains when the amount of labeled data is sparse.

Consistency regularization constrains the predictions in the neighborhood of a sample to produce the same prediction as the unperturbed sample, and is based on the validity of the cluster assumption. The simplest way to enforce consistency regularization is to perform random perturbations to the input. However, the space of possible perturbations in high-dimensional data is very large and randomly choosing perturbations can be ineffective. The ICT approach uses an interpolation between a pair of unlabeled samples such that the new sample lies in between the two, and consistency is enforced by constraining the transition curve between the two points is linear.

CutMix (Yun *et al.*, 2019) trains classifiers by introducing a random rectangular mask in the image to inpaint another image, and effectively combine the two images to produce a perturbation. Geoff et. al (French *et al.*, 2019) combines CutMix with the Mean Teacher (MT) model (Tarvainen and Valpola, 2017), and predict pixel-wise semantic segmentation in a semi-supervised fashion.

Adversarial training is another method used to enforce consistency. In this family, the objective is to train a discriminative model alongside a generative model. (Zhang *et al.*, 2017b) uses a Deep Adversarial Network (DAN) to train an evaluator network to discriminate between the segmentation produced from labeled and unlabeled images. The main segmentation network generates outputs that decrease the ability of the discriminator to distinguish between labeled and unlabeled samples. Another work on

adversarial training is the Virtual Adversarial Training algorithm (VAT) proposed in (Miyato *et al.*, 2019), which enforces consistency in the output with respect to adversarial input perturbations. The method replicates the effect of consistency regularization, while also being robust to adversarial attacks.

Deep Co-Training (DCT) (Qiao *et al.*, 2018) is another method to regularize outputs to stay consistent despite augmentations of the input, by minimizing Shannon divergence between the representations of the two views. Uncertainty-guided Cross-head Co-Training (UCC) (Fan *et al.*, 2022) uses a multi-head decoder that processes both strongly and weakly augmented data following the concepts outlined in (French *et al.*, 2019), but also uses uncertainty to filter out low-confidence pseudo-labels.

This work takes a similar approach to (Fan *et al.*, 2022; Sime *et al.*, 2022), where the decoder uses a multi-branch architecture and use the mean-teacher model to distill knowledge between the central student decoder branch and the auxiliary teacher decoder branches. This work differs from these approaches in that it performs an interpolation between the representations at the final layer of the encoder for the unlabeled samples to perform consistency regularization.

6.3 Methodology

This section provides a detailed breakdown of the method’s components and the rationale behind the components of the model used in this method.

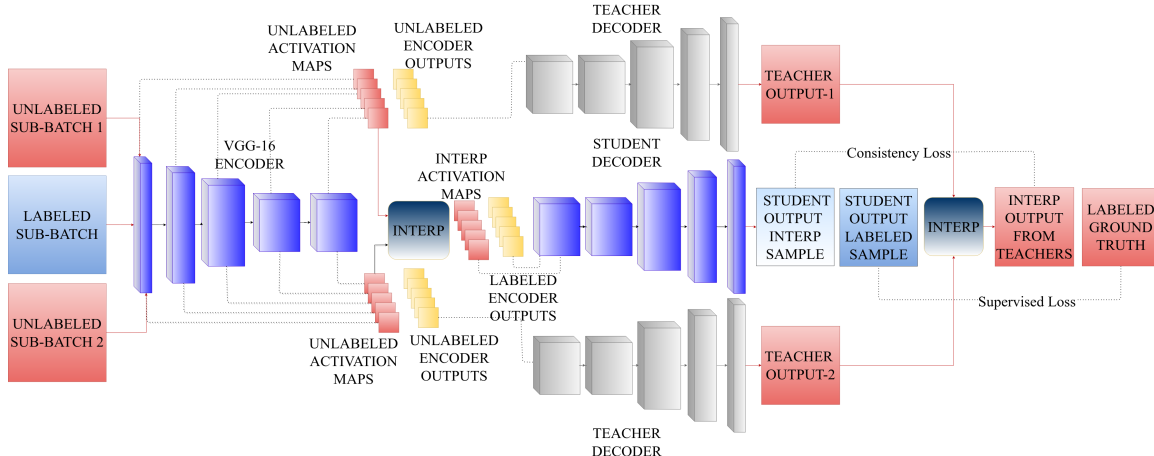


Figure 27. Overall methodology for semi-supervised segmentation using interpolation consistency in the probability space: Blocks with the same colors denote related elements. Note that the blue box denotes the common decoder weights for both the labeled data and the interpolated representation from the unlabeled pair. The training is to be done concurrently for both labeled and unlabeled batches.

6.3.1 Overview of the Proposed Framework

The proposed method takes in a corpus of a small set of labeled images and a larger set of unlabeled images. The samples are passed into a multi-branch network, where a multi-component loss is learned jointly. Labeled samples are learned using a supervised loss, and unlabeled samples are learned using a consistency loss.

Let D_L be the set of labeled training samples, and D_U be the set of unlabeled training samples. Let (x_L, y_L) be a pair of labeled samples from D_L , and let (x_i, x_j) be a pair of unlabeled samples extracted from D_U . Detailed batching schemes for the unlabeled samples is given in Section 6.3.3. The training process then transforms these labeled and unlabeled samples using horizontal and vertical flipping, affine rotation and scaling, and color jitters. These transforms increase the diversity of the samples by perturbing it around the neighborhood.

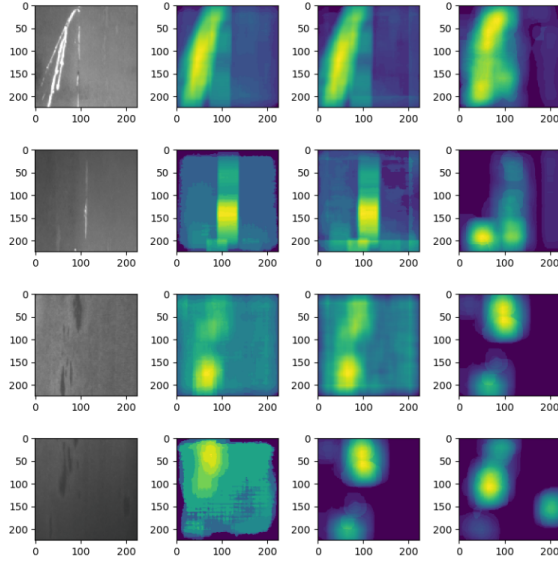


Figure 28. Cluster assumption demonstration by computing the patchwise Euclidean distance between a patch and its neighbors. Higher values are indicated with yellow: Left-most image is the input, followed by the distance maps of the following: the input image, the activation map from layer 4, and the activation map from the final layer of the encoder.

The labeled sample is drawn from the joint distribution $P(X, Y)$, and is treated exactly the same as it is in fully supervised segmentation with encoder-decoder type architectures. The unlabeled sample is also passed through the common encoder branch. The unlabeled samples are not augmented versions of the same sample, but two different samples that are drawn from the same distribution of the data generating process $P(X) = \frac{P(X, Y)}{P(Y|X)}$. The consistency task in this approach is not to enforce the same prediction for two augmented samples as is seen in the image classification case. Rather, the aim is to enforce consistency on interpolated versions of the unlabeled samples in the latent space. The justification for this approach lies in the cluster assumption. Images at the input level are high dimensional representations that do not strictly obey the cluster assumption, as argued in (French *et al.*, 2019). The hidden representations, however, are at a lower dimension, and were expected to have

better clustering properties. This is identified by plotting the hidden representations’ patch-wise Euclidean distances, by first obtaining a class activation map and then interpolating it to the original image size, as shown in Figure 28. The patch-wise distances better conform to object boundaries at the hidden representation level, as compared to the input level.

Let (x_i^{CAM}, x_j^{CAM}) be the probability maps of the representation in the latent space. The class activation maps are obtained using a global average pooling (GAP) layer (Zhou *et al.*, 2016). The GAP performs a channel-wise spatial averaging of data, and this is multiplied by the weights w_k^c to yield the activation map M_c . This is then converted into a channel-wise probability by using the softmax:

$$m^{CAM}(i, j) = \sum_k w_k^c \cdot f_k(i, j) \quad (6.1)$$

$$x^{CAM}(i, j) = \text{Softmax}(m^{CAM}(i, j)) \quad (6.2)$$

The difference between this approach and the one described in ICT (Verma *et al.*, 2019) is that the interpolation happens at the latent space, enforcing consistent predictions during interpolations between representations of the unlabeled data. To interpolate, a convex combination of the two unlabeled samples is first obtained, represented as $x_{mix} = \lambda \cdot x_i^{CAM} + (1 - \lambda) \cdot x_j^{CAM}$, where λ is a mixing coefficient sampled from the beta distribution, $\text{Beta}(\alpha, \alpha)$, to generate interpolated data points. Consistency is then enforced between the decoded representation from this interpolated map, and the decoded representation from the teacher decoders:

$$f_\theta(\text{Mix}_\lambda(x_i^{CAM}, x_j^{CAM})) \approx \text{Mix}_\lambda(f_{\theta'}(x_i^{CAM}), f_{\theta'}(x_j^{CAM})) \quad (6.3)$$

Here, f_θ represents the model being trained. while θ' is an exponential moving average

(EMA) of θ , given by:

$$g(\theta_t) = \gamma g(\theta_{t-1}) + (1 - \gamma)f(\theta_t) \quad (6.4)$$

$$\gamma = \min\left(1 - \frac{1}{n + 1}, \gamma\right) \quad (6.5)$$

Here, n is the number of iterations completed and γ is a weighting factor. This dynamic weighting process ensures that the EMA starts by being dominated by the second term, eventually converging to a stable computation using γ as the weighting factor.

6.3.2 Loss Functions

The consistency between the interpolated outputs is enforced using the KL divergence between the prediction given by the latent-space interpolation \tilde{y}_{interp} and the prediction from the interpolation that takes place after each unlabeled sample is decoded, $\text{interp}(f_\theta(x_i), f_\theta(x_j))$.

$$L_{cons} = \sum_i \tilde{y}_{interp} \log \left(\frac{\tilde{y}_{interp}}{\text{interp}(f_\theta(x_i), f_\theta(x_j))} \right) \quad (6.6)$$

The supervised loss function is a combination of a distributional loss and two shape-aware losses. The distributional loss used for this work is the focal loss (Lin *et al.*, 2018), that takes into account the class imbalances present in the training data. This loss function weights misclassified training samples by introducing a focusing factor $\gamma \geq 0$. This study uses $\gamma = 2$. Additionally, the parameter α is used to weight the classwise loss using median-frequency balancing:

$$L_{focal}(p_t) = -\alpha(1 - p_t)^\gamma \log(p_t) \quad (6.7)$$

$$\alpha_c = \frac{f_c^m}{f_c}$$

f_c^m is the median frequency of the class across the sample set and f_c is the class frequency. The next two components of the loss are optimized to improve the prediction boundary directly, consisting of the IoU loss (Berman *et al.*, 2018), L_{IU} and an active boundary loss (ABL), L_{ABL} , whose details of computation are described in Chapter 4. The total loss is then:

$$L = L_{focal} + L_{IU} + L_{ABL} + w(t)L_{cons} \quad (6.8)$$

6.3.3 Dataset Batching

In this section, the implementation of the mini-batch sampler is detailed. It is designed to manage sampling for both labeled and unlabeled data. It is initialized with parameters such as the percentage of labeled data and batch sizes for labeled and unlabeled data. The mini-batch sampling process involves several stages:

1. **Shuffling:** The labeled and unlabeled image lists are shuffled to ensure random sampling.
2. **Sub-batch Creation for Unlabeled Data:** The unlabeled data is divided into sub-batches of a pre-defined size. Each sub-batch is further segmented to obtain pairs of unlabeled data for each instance of a labeled sample.
3. **Random Sampling of Labeled Data:** Labeled data is randomly sampled to create chunks corresponding to the number of unlabeled sub-batches.
4. **Combination of Labeled and Unlabeled Chunks:** The sampler then combines labeled and unlabeled chunks into composite batches, ensuring a mix of both data types. This step adapts to scenarios with imbalanced counts of labeled and unlabeled samples, with the caveat being that the unlabeled batch size needs to be even for the sub-batch creation.

5. **Index Conversion:** Finally, the batches are converted into indices based on their positions in the dataset, preparing them for extraction by the Dataset class.

6.4 Experiments

Experiments were conducted on defect datasets from three different domains and two different imaging modalities. The first dataset is the Northeastern University (NEU) metallic surface defect dataset (Dong *et al.*, 2019) which has the largest number of samples and has 3 defect categories that form in strip steel plates: patches, scratches and inclusion. Patch defects have a darker color than the background steel material and scratch defects have lighter colored segments compared to the background. Inclusion defects present lower contrast compared to the patch defect, and are caused due to oxidation. The second dataset is the Magnetic Tiles (MagTile) Dataset (Huang *et al.*, 2018), which is a smaller dataset with 5 defect categories: Blowholes, crack, break, fray and uneven. Among these defects, the uneven and crack defect categories blend well with the background, making segmentation challenging. The MagTile dataset also presents with some inter-class similarities, such as between the break and the blowhole class. The intra-class variance in the crack and fray defects also makes this dataset challenging. Figure 29 and Figure 30 show some samples from the MagTile and NEU datasets respectively. The training and test configurations are given in Table 11.

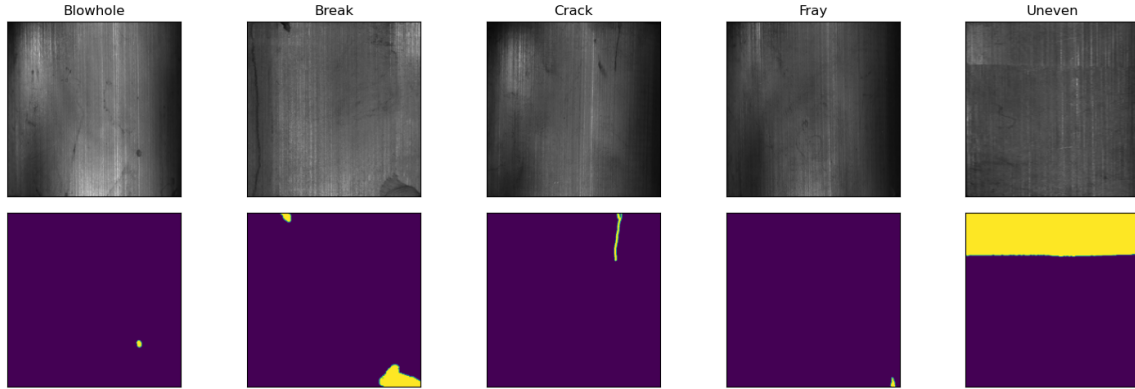


Figure 29. Demonstrative images from the Magnetic Tiles defect dataset

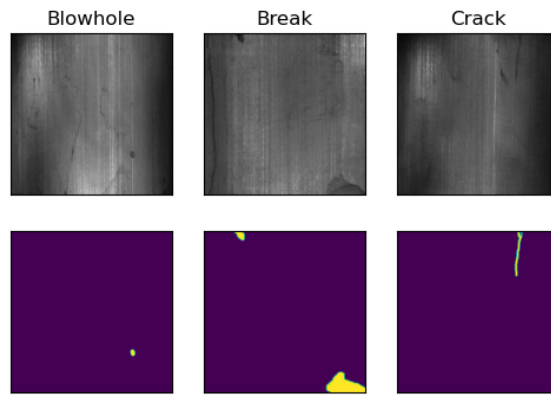


Figure 30. Demonstrative images from the NEU steel surface defects dataset

Dataset	Training	Test
NEU-Seg	3088	840
MagTile	312	80

Table 11. Dataset sizes

6.4.1 Network Setup and Training

The overall network architecture consists of an encoder, and two decoders, the teacher decoder and the student decoder. The encoder is a VGG-16 network pretrained on the ImageNet dataset. The final fully connected layer and average pooling layer

are removed and replaced with an adaptive average pooling layer. The fully connected layer is replaced with a 1x1 convolutional layer at the output of each of the 5 sub-blocks of the network, used to compute class activation maps at each stage. The decoder consists of a series of transposed convolutional layers, to upsample the image back to the original size. The dataset is loaded into the network using a batching method that provides 3 sub-batches for every iteration: 1 sub-batch for labeled images, and the other two consist of unlabeled images. The labeled images are iteratively sampled with replacement until all the unlabeled images are accounted for. The labeled image passes through the common encoder and the student decoder. The unlabeled image first passes through the common encoder, and the corresponding class activation maps are extracted and normalized to sum to 1 in the class dimension. Each of the 5 activation maps from the unlabeled data pair is interpolated, and the interpolated activation maps are passed into the student decoder, and resized back into the right channel dimensions using a 1x1 convolution. Refer to the below provided layer details tables (Table 12 - 14) for information about the layers in the model.

The remaining unlabeled samples are passed through the teacher decoder and interpolated after the outputs are obtained. The consistency loss is computed between the output obtained from the interpolated inputs through the student decoder and the output interpolated after they pass through the teacher decoder.

Table 12. Encoder Layer Architectural Details with Channels Subdivision and Operation Parameters

Layer	Channels		Parameters			Output
	Input	Output	Kernel	Stride	Padding	
Conv2d	3	64	(3, 3)	(1, 1)	(1, 1)	224×224
Conv2d	64	64	(3, 3)	(1, 1)	(1, 1)	224×224
MaxPool2d	-	-	(2, 2)	-	0	112×112
Conv2d	64	128	(3, 3)	(1, 1)	(1, 1)	112×112
Conv2d	128	128	(3, 3)	(1, 1)	(1, 1)	112×112
MaxPool2d	-	-	(2, 2)	-	0	56×56
Conv2d	128	256	(3, 3)	(1, 1)	(1, 1)	56×56
Conv2d	256	256	(3, 3)	(1, 1)	(1, 1)	56×56
Conv2d	256	256	(3, 3)	(1, 1)	(1, 1)	56×56
MaxPool2d	-	-	(2, 2)	-	0	28×28
Conv2d	256	512	(3, 3)	(1, 1)	(1, 1)	28×28
Conv2d	512	512	(3, 3)	(1, 1)	(1, 1)	28×28
Conv2d	512	512	(3, 3)	(1, 1)	(1, 1)	28×28
MaxPool2d	-	-	(2, 2)	-	0	14×14
Conv2d	512	512	(3, 3)	(1, 1)	(1, 1)	14×14
Conv2d	512	512	(3, 3)	(1, 1)	(1, 1)	14×14
Conv2d	512	512	(3, 3)	(1, 1)	(1, 1)	14×14
MaxPool2d	-	-	(2, 2)	-	0	7×7

6.4.2 Prediction

The trained model is evaluated using the intersection over union (IoU) metric, commonly used in segmentation tasks. IoU measures the degree of overlap between the ground truth and the predicted image mask - a score of 1 indicates a perfect match.

$$IoU = \frac{\text{Area of Overlap}}{\text{Area of Union}} = \frac{TP}{TP + FP + FN} \quad (6.9)$$

where TP is the number of true positive pixels (pixels classified as positive by both the model and the ground truth), FP is the number of false positive pixels (pixels

Table 13. CAM Block Details

Layer	Input Channels	Output Channels	Kernel
AdaptiveAvgPool2d	-	-	(1, 1)
Conv2d-BLOCK 5	512	N_{class}	(1, 1)
Conv2d-BLOCK 4	512	N_{class}	(1, 1)
Conv2d-BLOCK 3	256	N_{class}	(1, 1)
Conv2d-BLOCK 2	128	N_{class}	(1, 1)
Conv2d-BLOCK 1	64	N_{class}	(1, 1)

Table 14. Decoder Architecture

Layer	Channels		Parameters			Output
	Input	Output	Kernel	Stride	Padding	
ConvTranspose2d	512	512	(3, 3)	(2, 2)	(1, 1)	14×14
ConvTranspose2d	512	256	(3, 3)	(2, 2)	(1, 1)	28×28
ConvTranspose2d	256	128	(3, 3)	(2, 2)	(1, 1)	56×56
ConvTranspose2d	128	64	(3, 3)	(2, 2)	(1, 1)	112×112
ConvTranspose2d	64	32	(3, 3)	(2, 2)	(1, 1)	224×224
ConvTranspose2d	32	4	(1, 1)	(1, 1)	(0, 0)	224×224

classified as positive by the model but negative by the ground truth), and FN is the number of false negative pixels (pixels classified as negative by the model but positive by the ground truth). Model uncertainty is obtained using Monte-Carlo (MC) Dropout. MC Dropout is a Bayesian deep learning method used for estimating uncertainty in neural networks by activating weights using the Bernoulli distribution with probability p . During prediction, dropout is activated and the network is run multiple times with different dropout masks to produce a distribution of predictions. This distribution can then be used to estimate the model’s uncertainty and confidence in its predictions.

6.5 Results and Discussion

This section presents the results obtained on the NEU surface defect dataset and the Magnetic Tile (MagTile) dataset. The larger NEU dataset is used for the bulk of the evaluations due to the larger sample size. First, the effect of adding unlabeled data on model predictive performance is shown as compared to just using labeled data. Loss function weighting between the consistency and supervised losses is also studied, by using a sigmoid ramp-up function for the consistency loss. The results show that the mIU obtained in the validation set using the semi-supervised approach exceeds the fully supervised baseline, as shown in Table 15. The level of performance improvement obtained by the inclusion of the unlabeled data is the most dramatic when the amount of labeled data is scarce. This gap reduces as more labeled data is available, with the performance converging toward the fully-supervised baseline. To test how much better the addition of 30 labeled samples is to the model, the model was trained with only unlabeled samples. This meant that the network did not have any supervised loss component, but only an unsupervised loss. This resulted in a poor mIU of 8% which was improved more than 7x when a limited supervised set of 30 samples was added. This result shows that the proposed method can reduce image-labeling efforts.

Figure 33 shows some sample detections while using only 30 labeled images. The demonstrations indicate that the semi-supervised learning model manages to obtain correct detections in places where there are none in the supervised case, and also manages to eliminate some spurious noise produced in the detections of the supervised model. Figure 32 summarizes a compilation of detections in the validation set with various amounts of labeled data.

Labeled samples	Supervision	mIU
0%	Consistency Loss Only	0.08
1%	Supervised	26.87
	Semi-Supervised	59.17
10%	Supervised	59.75
	Semi-Supervised	78.22
30%	Supervised	74.21
	Semi-Supervised	80.58
50%	Supervised	79.97
	Semi-Supervised	82.53

Table 15. Effect of incorporating unlabeled samples at different ratios to labeled samples in the NEU test set

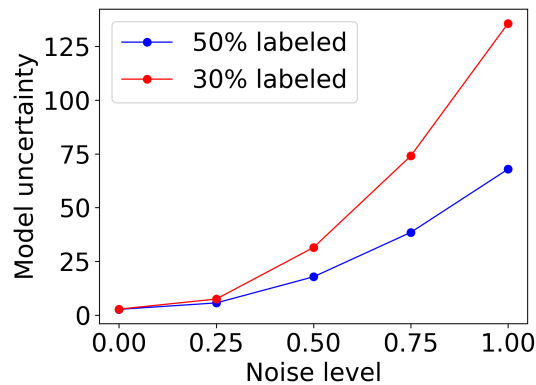


Figure 31. NEU test-set output uncertainty as a function of input Gaussian noise level

A comparison was made between equally weighted losses and a ramp-up function applied to the consistency loss over a number of epochs. In this case, the number of epochs was chosen to be 80 empirically, using the evolution of the supervised loss function in the unweighted case. This made the supervised loss component dominant in the initial stages of the training. However, the performance produced by this approach was consistently lower on the NEU dataset than that produced when both the losses were weighted equally, as shown in Figure 34.

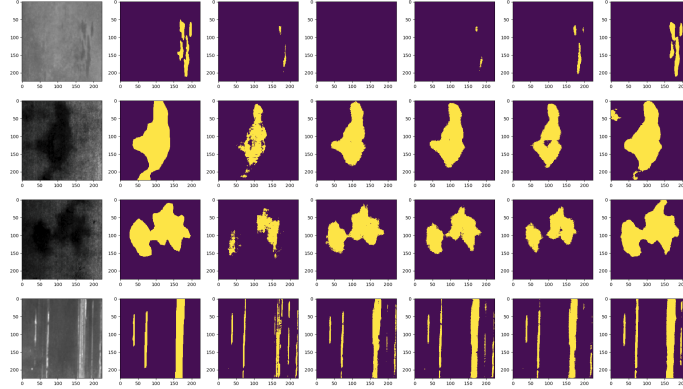


Figure 32. Demonstrative detections on the NEU validation set: (Left to Right) Image, Ground Truth, 5% Labeled, 10% Labeled, 20% Labeled, 50% Labeled, Fully Labeled.

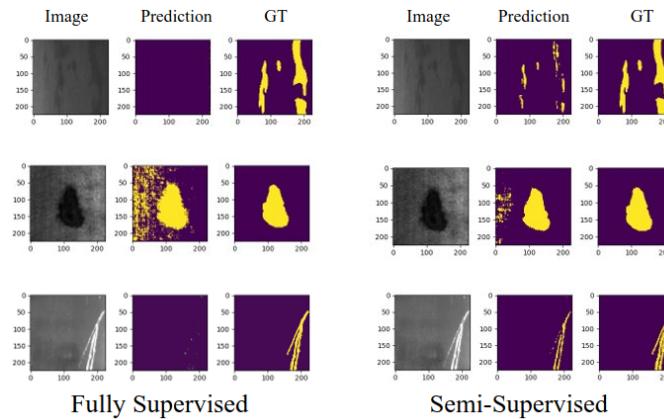


Figure 33. Demonstrative detections illustrating the effect of adding additional unlabeled data to the set and using the consistency loss. In this case, the comparison is made between the fully-supervised case with 30 labeled samples, and the semi-supervised case with 30 labeled samples and 570 unlabeled samples.

The results obtained from the proposed method were then compared against benchmark results from recent works in Table 16, and is shown to outperform contemporary methods for most cases. In the NEU dataset, the model under-performs state of the art at the 1% labeled level. An ablation study on the supervised losses is performed by removing the boundary loss during training. This shows that the model gets

significant benefits from the boundary loss, with the results degrading up to 12.5% on the NEU dataset.

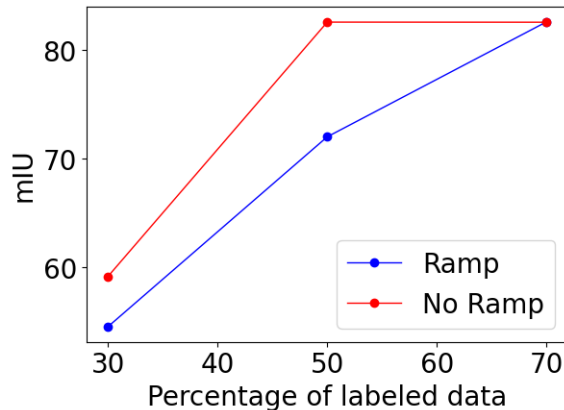


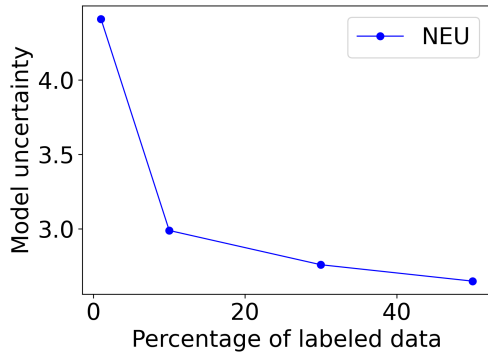
Figure 34. Test-set performance on the NEU test set with and without consistency loss ramp-up

Method	NEU				MT
	1%	10%	30%	50%	40%
DAN (Zhang <i>et al.</i> , 2017b)	65.74	73.79	75.76	76.63	55.72
DCT (Qiao <i>et al.</i> , 2018)	66.71	74.05	76.97	78.49	64.46
MT (Tarvainen and Valpola, 2017)	67.62	73.49	76.80	78.16	63.78
CCT (Ouali <i>et al.</i> , 2020)	67.78	74.56	77.38	77.77	59.45
ICT (Verma <i>et al.</i> , 2019)	68.29	74.94	76.24	77.74	60.60
CPS (Chen <i>et al.</i> , 2021)	68.19	74.98	77.94	78.98	60.77
UCC (Fan <i>et al.</i> , 2022)	<u>69.06</u>	74.30	76.67	77.63	62.18
ReCo (Liu <i>et al.</i> , 2021)	68.83	75.11	77.01	78.10	64.10
SimCEPS (Sime <i>et al.</i> , 2022)	70.36	<u>75.94</u>	<u>78.94</u>	<u>79.95</u>	65.28
CRAMI (Focal + ABL-IU)	59.17	78.22	80.58	82.53	<u>71.22</u>
CRAMI (Focal + IU)	52.98	70.14	73.04	70.01	71.31

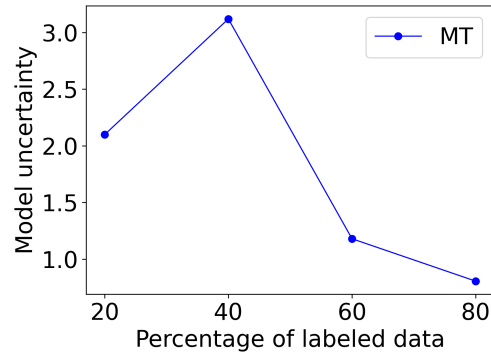
Table 16. Results for the NEU and MagTile datasets compared against similar models

Finally, the effect that the labeled samples have on the model uncertainty is shown. Robustness of the model predictions to corruptions in the input is also shown as a function of the labeled information available during training. Figure 35b shows that

for the NEU dataset, there is a consistent reduction in uncertainty with the addition of more labeled samples. For the MagTile dataset in Figure 35b, while there is a general trend of uncertainty reduction, there is an outlier at the 40% uncertainty level, where the uncertainty increased.



(a) NEU Dataset - Model Uncertainty



(b) MagTile Dataset - Model Uncertainty

Figure 35. Model uncertainty as a function of percentage of labeled data

6.6 Conclusion

This chapter presents a method to perform semi-supervised training for semantic segmentation using activation map interpolation. The method takes in a set of labeled and unlabeled batches, interpolates the unlabeled batches' activation maps, and then uses interpolation consistency regularization to minimize the difference between the output of the prediction from the interpolated input. The network has an encoder-decoder style architecture and a student-teacher style decoder. The prediction from the unlabeled inputs interpolated after obtaining their respective predictions. The model also uses an Exponential Moving Average (EMA) scheme to update the parameters of the teacher decoder, while performing backpropagation on the student decoder. The study shows improvements to model performance against existing benchmarks on the NEU metal surface defect and MagTile datasets. The results indicate a consistent improvement in performance compared to purely supervised learning at low sample regimes, and this difference in performance narrows down as more data is added.

UNCERTAINTY-AWARE DECISION SUPPORT USING ON-BOARD VISION DATA FOR PILOT SITUATIONAL AWARENESS

7.1 Introduction

Over the past two decades, the aviation industry has made remarkable strides in safety and automation. Air transport is broadly divided into two categories: General Aviation (GA) and Commercial Aviation (CA). GA is typically utilized for private flying and recreational travel, whereas CA encompasses scheduled passenger and cargo flights run by airlines. Despite the overall advancements in aviation safety, GA operations still fall behind in terms of safety standards compared to their commercial counterparts (Wilson and Sloan, 2003). This is because GA operations are subject to looser regulations, greater variation in on-board equipment, and more frequent operations in hazardous conditions with terrain and weather risks. Commercial operations usually have highly regulated, consistent operational and safety standards, with compliance being given great emphasis. These factors ensure that commercial operations have a more cohesive and strongly enforced operational procedure set. On the other hand, GA pilots often fail to adhere to such strict procedures, leading to cascading outcomes from poor decisions (Guo *et al.*, 2021b). Additionally, sensing equipment on-board commercial aircraft is often standardized, with modern commercial jets having advanced sensing equipment like weather and terrain radar. Smaller GA aircraft, particularly older piston engine types, lack advanced sensors to detect weather and terrain threats. These are not required and are usually costly to obtain. Nevertheless,

GA pilots frequently use software to obtain real-time weather data from sources like the Next Generation Weather Radar (NEXRAD) data. Weather radar is expensive and requires specialized tuning to detect different types of precipitation and clouds. Moisture levels within the clouds also significantly affect signal reflection.

Among the various threats that a GA pilot might encounter is the inadvertent loss of situational awareness. Situational Awareness (SA) is paramount for flight safety - Loss of SA has been documented to be a significant contributor to incidents. Accumulated fatigue and multi-tasking during critical phases of flight often exacerbate this risk. A key aspect of SA that this chapter will focus on is the loss of SA due to inadvertent entry into instrument conditions. This remains a challenge that private pilots encounter in both VFR and IFR operations, leading to threats such as encountering icing conditions(Cao *et al.*, 2018), violation of Visual Flight Rules (VFR) limits, and an increased risk of Controlled Flight Into Terrain (CFIT) (Kelly and Efthymiou, 2019; Majumdar *et al.*, 2021). Fatal accidents due to entry into Instrument Meteorological Conditions (IMC) have a risk tolerance and incorrect risk perception component, as indicated in (Madhavan and Lacson, 2006). These accidents are also caused by a lack of prior knowledge about developing weather conditions and insufficient pre-flight planning. Continuous autonomous monitoring of sky conditions and a proper synthesis of the monitored data in a decision-support framework can therefore serve to augment the capacities of the pilot. Addressing the limitations in weather-related decision-making training, Johnson and Wiegmann (2015) underscored the value of simulation in improving pilots' skills. While simulations cannot replicate all factors, they provide a controlled environment to practice critical decisions safely.

To improve GA operational safety, computer vision based approaches are used to complement existing methods and data sources and propose an uncertainty-aware

decision support system that uses image data from on-board cameras. The proposed system provides many benefits, such as cheap installation costs, operating costs and the ease of integration into existing decision support systems.

As part of the efforts to achieve data-driven, vision-based decision support, a novel system that utilizes camera data to understand the scene is proposed. Image features, weather features and sentence-based weather descriptors are combined to produce label-dependency aware predictions. Another key aspect of this chapter is the discussion on the choice of variables that are used for prediction - the focus is on real-time prediction of threats that can result in entry into Instrument Meteorological Conditions (IMC). Since pilots are trained to read Meteorological Aerodrome Report (METAR) reports, this approach seeks to build a model that can predict threats that are mentioned in METAR reports to provide a unified threat representation.

The contributions of this chapter are:

1. A multi-label, multi-class learning approach for decision support in general aviation with the ultimate goal of reducing inadvertent entry into IMC by employing the use of image data and weather data from METAR tables.
2. A benchmark dataset of images for training this threat prediction model that is acquired from state-of-the-art flight simulation software.

The rest of this chapter is organized as follows: Section 7.2 discusses the state of the art in decision support and threat prediction systems for GA applications. Section 7.3 expands upon the problem setup and model architecture proposed in this work. Section 7.4 provides details on the dataset and how it was collected, the training and evaluation setup. Section 7.5 provides detailed results of the model performance, followed by a discussion on the limitations of this work. Section 7.6 summarizes the contribution of this work and some future directions.

7.2 Related Work

7.2.1 Decision Support Systems in Aviation

Decision support systems have seen significant upgrades in Air Traffic Management. Data-driven predictive systems have recently been used for applications such as aircraft re-routing (Zhang and Mahadevan, 2017) and trajectory prediction (Pang *et al.*, 2022; Zhang and Mahadevan, 2020; Pang *et al.*, 2021; Ciccio *et al.*, 2016). However, Predictive systems are not as prevalent in GA operations. Painter *et al.* (1997) used the concept of an "anticipatory" Flight Management System (FMS) for GA pilots. This early work emphasized the integration of personal computing software into flight management architecture, thereby enhancing pilot situational awareness. By detecting flight phases and displaying pertinent information, the system alleviated information overload. Although rooted in the commercial aviation domain, the integration of advanced FMS principles into GA operations laid the foundation for more adaptive decision support systems for more sophisticated GA aircraft. In a more recent study (Whitehurst *et al.*, 2019), the challenges associated with interpreting aviation weather products were highlighted. Pilots often struggle to decipher weather-related cues from available products, even those with intermediate training and experience. The chapter recommended the development of a decision support tool to dynamically guide pilots in weather-related decision-making. This recommendation resonates with the proposal of this study, which aims to address the difficulties in weather interpretation by offering real-time decision guidance. Johnson *et al.* (2017) investigated the latency of weather data from portable electronic devices. This study emphasized the need for real-time monitoring solutions, which aligns with the proposed approach of a decision support

system capable of providing timely predictions. The fusion of synthetic and enhanced vision systems to improve visibility during flight operations was examined by Kramer *et al.* (2009). While primarily focused on commercial aviation, the study sheds light on challenges related to scalability and cost, factors relevant to implementing such systems in GA operations. Tippey *et al.* (2017) conducted research on the effects of weather technology interface characteristics on decision-making quality and timeliness. The study found that vibrotactile cues improved reception of notifications and reduced cognitive demands during flight, reinforcing the significance of incorporating effective visual and tactile display characteristics into decision support tools for GA pilots.

7.2.2 Threat Prediction Models in General Aviation

Monitoring systems have played an important role in autonomous navigation in unmanned aerial vehicles (UAVs), traffic collision avoidance systems (TCAS), and more recently, implementations of synthetic vision and vision-based navigation tools in some modern aircraft. In the realm of threat monitoring and avoidance systems, several recent papers contribute valuable insights. (Opromolla and Fasano, 2021) introduces a vision-based approach for obstacle detection and tracking in small Unmanned Aircraft Systems (UAS), employing deep-learning neural networks to assess collision threats. (Ramasamy *et al.*, 2016) presents a LIDAR-based threat avoidance and trajectory proposal system for UAS in urban areas. Vivo *et al.* (2021) addresses the challenge of wildfire monitoring using UAVs and proposes a data-driven edge detection algorithm for fire front detection, even in low visibility conditions. Jeong *et al.* (2021) presents a deep neural network-based hazardous flight region prediction system for small UAVs in urban areas, considering wind environments and local topography to assess flight

hazards. Pang *et al.* (2022) proposes an uncertainty-aware multi-aircraft trajectory prediction model for the near terminal airspace for commercial aviation using a deep neural network approach. In the domain of aviation trajectory prediction with a focus on weather uncertainties, an additional noteworthy contribution is made by Pang *et al.* (2021) that presents a Bayesian deep learning approach for trajectory prediction considering weather data and uncertainties. These papers collectively emphasize the potential of deep learning and data-driven algorithms in enhancing situational awareness and safety. Nearly all of the work on such monitoring systems is either for highly regulated commercial aviation operations or for UAS. Arguably, GA operations are closer to UAS when it comes to ubiquity and scale, with the potential for faster adoption of new techniques. To my knowledge, this is the first study to introduce a framework for real-time assessment of weather situations for general aviation pilots that considers weather information and camera image data using a multi-task learning approach. This work would serve as an important starting point for future works that build on this single-agent formulation.

7.3 Methodology

7.3.1 Overview

The proposed framework uses a multi-label, multi-class learning approach inspired from the work by Chen *et al.* (2019), as the goal is to learn a model that can not only learn image features, but also learn co-occurrences of these features in a multi-label setting. The image encoder used in deep learning methods has been a consistent component of multi-label classifiers. However, the usage of a graph neural network

module to learn label co-occurrences has been proposed by Chen *et al.* (2019), where they used the label co-occurrences in the ground truth of the image dataset to model label co-occurrences. This idea is expanded upon to hypothesize that there may be a benefit to using other data sources to model these co-occurrences, and learn to balance the information from the image representation and the other sources. Given the benefits that augmentation produces, an ensemble of these representations is proposed to improve prediction performance. This ensemble approach aims to integrate diverse data sources effectively, thereby enhancing the model’s ability to make accurate predictions in complex, real-world scenarios such as those encountered by GA pilots entering IMC conditions.

7.3.2 Problem Setup

The proposed approach casts weather threat detection as a multi-label, multi-class classification problem. The model takes an input image $X \in R^{(W \times H)}$ and returns a 14-dimensional prediction vector $Y \in R^{(1 \times 14)}$. The prediction is divided into 3 separate heads - The weather threat prediction head (WT), the cloud cover classifier head (CC), and the meteorological condition classifier head (MC). Next, each of these target variables are described in detail with an explanation of their significance to improving SA for GA pilots.

7.3.2.1 Weather Threats Prediction

The WT head predicts the presence of Fog (FG), Mist (BR), Precipitation (SN/RA), and the threat of entry into the cloud if the pilot continues to fly at the present altitude

(TEC-PA). TEC-PA is the only variable that is not present in METAR reports. The proposed model aims to predict this, as the goal is to capture the localized weather representation with the model. Given that the current implementation of this work only uses RGB data, the lack of direct depth information may be somewhat alleviated if one can predict this variable in a supervised manner. Similarly, the FG and BR conditions are also defined in the ground truth dataset to be consistent with the meteorological definitions specified by the International Civil Aviation Organization (ICAO). Fog and mist are similar weather phenomena that ultimately reduce ambient visibility due to suspended water droplets in the atmosphere. Fog is a more extreme visibility restriction, with visibility less than half a mile. Mist is defined as reduced visibility due to suspended water droplets, with visibility ranging between half a mile to 5 miles. Precipitation prediction is also important for aviation safety, as these can be major sources of icing and indicate a high risk of entry into IMC conditions. Precipitation shafts, in particular, are special phenomena that can serve as indicators of convective phenomena that present risks for aircraft upsets.

7.3.2.2 Cloud Cover Classification

A novel aspect of the proposed model is the utilization of detailed visual data from the camera to predict cloud cover as seen from the aircraft's perspective. This approach is justified based on the following rationale: METAR data, while offering cloud cover information in oktas, cloud base altitudes, and updates every few minutes, is limited to airfield locations. NEXRAD provides comprehensive top-down satellite views, including data on convective layers, precipitation, and cloud tops. However, the proposed system aims to bridge the gap by enabling continuous and potentially

multi-view monitoring of cloud cover directly from the aircraft. This offers a unique vantage point for obtaining descriptors of cloud morphology.

While the scope of this study is limited to predicting a consolidated representation of cloud cover, future work could extend to segmenting the sky for more detailed predictions, thereby enhancing the precision of cloud cover assessments. It is crucial to clarify that the METAR data referenced here does not influence how label co-occurrences are computed. Instead, it pertains to the type of information available to pilots in-flight.

7.3.2.3 Instrument Meteorological Condition Prediction

Finally, the MC head in this model is a classifier that divides the prediction into either Instrument Meteorological Conditions (IMC), Marginal Visual Meteorological Conditions (M-VMC), or Visual Meteorological Conditions (VMC). These meteorological categories are classified based on ceiling and visibility. The International Civil Aviation Organization (ICAO) defines the ceiling as "the altitude above ground level (AGL) of the lowest cloud base that is below 20,000 ft covering more than half the sky." The FAA defines visibility to be "the furthest distance at which prominent objects can be viewed with the naked eye." The lack of direct-depth information challenges the accurate labeling of the ground truth. This can be accounted for during data collection, as described in Section 7.4. The definitions adopted for these categories are as follows:

1. IMC: Cloud ceilings within 1000 feet, and visibility below 3 NM.
2. M-VMC: Cloud ceilings between 1000 and 3000 feet, and visibility between 3-5 NM.

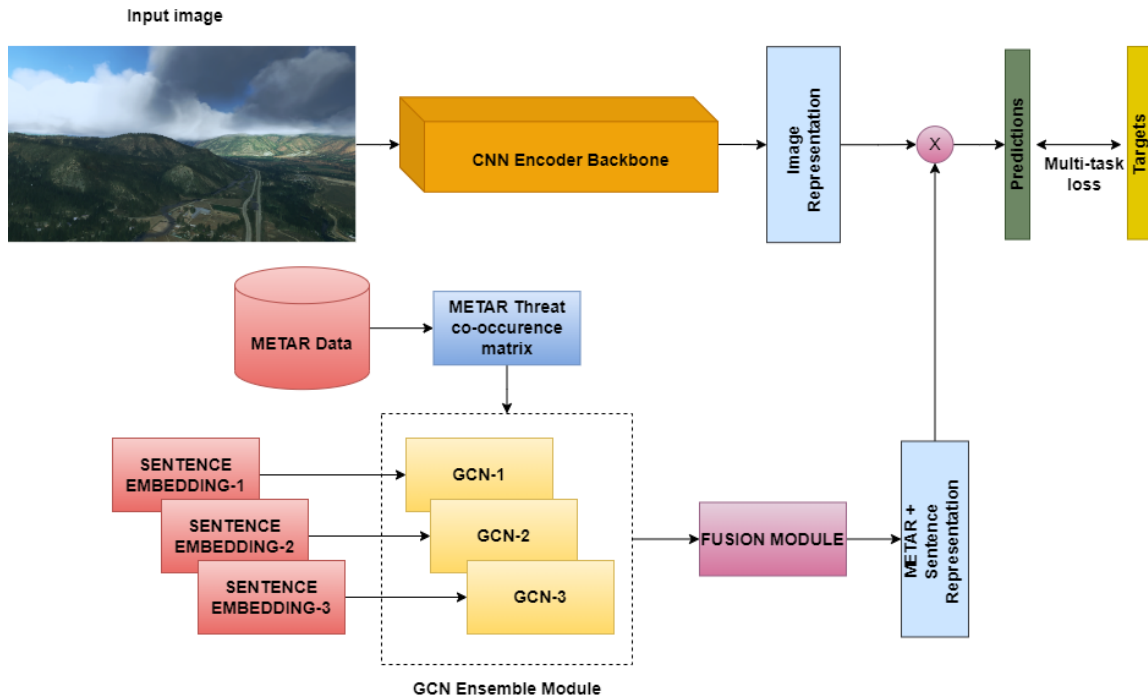


Figure 36. Overall network architecture for weather threats prediction

3. VMC: Cloud ceilings above 3000 feet, and visibility above 5 NM.

7.3.3 Multilabel, Multiclass Classification Model for Threat Prediction

The goal of the model is to ultimately incorporate image, language, and METAR data to form a useful representation for predicting the target variables described in Section 7.3.2. The raw data inputs are transformed into a compressed representation using various encoding modules described below. An overall architecture diagram is shown in Figure 36

7.3.3.1 Image Encoder Module

The image data is compressed into a smaller dimensional representation using a series of convolutional layers. A convolution is a spatial operation on an image or

feature \mathbf{X} tensor with dimensions $C \times H \times W$, where C represents the number of input channels, H is the height of the feature map, and W is the width of the feature map. The convolutional layer consists of learnable weights, called kernels \mathbf{W} with shape $C' \times C \times K \times K$, where C' is the number of output channels, K represents the spatial size of the filter, and C is the number of input channels. The appropriate kernel weights are learned through backpropagation using gradient descent during training. The kernels perform a weighted sum over each block of the input feature, transforming it into an output with shape $C' \times H' \times W'$, where C' represents the number of output channels, H' is the height of the output feature map, and W' is the width of the output feature map.

Image encoders that utilize a series of convolutional layers are widely used for compressing this information into a learned feature descriptor. The baseline case used the ResNet-18 architecture for the image encoder (He *et al.*, 2016b), which consists of a sequence of convolutional layers and non-linear activation layers. Average pooling is done at the end of the resnet block to reduce the size of the learned representation. The ResNet backbone is commonly used for its advantages in avoiding vanishing gradients in very deep networks. The image is finally compressed to a 512-dimensional vector.

7.3.3.2 Graph Representation Learning Module

In this section, the overall Graph Convolutional Network (GCN) processing module for label co-occurrence modeling is described. The differences of this study from the work by (Chen *et al.*, 2019) is also highlighted. While both approaches use the GCN to model label co-occurrences, this study differs in a few important ways: First,

while (Chen *et al.*, 2019) uses the image database to mine co-occurrences, this study computes co-occurrences of weather threat features from historical METAR data. The rationale for doing this becomes apparent if one considers that mining correlations from the image training set is only valid when one has thousands of images, and that METAR co-occurrences captured from a large enough database provides physically-based information to model threat correlations. Second, features from sentence embeddings are used to model the targets instead of averaging word embeddings. Third, a cost matrix is utilized alongside a focal loss to learn to classify in a label-imbalanced scenario that requires the penalization of non-conservative predictions. The details of the METAR data and sentence embedding representations are discussed in Section 7.4.1.1, and further elaborations on the cost matrix and focal loss is done in Section 7.3.3.4.

The GCN module is used for operations on graph-structured data. Unlike images that are grid-like, a graph G can directly model relationships between different variables V using edges E to indicate the strength of the relationship between them. A graph convolution is a generalization of the convolution operation described in the previous section to nodes. Similar to how convolutional layers operate on features using kernel matrices, one can obtain the following for a GCN layer: Let $h_v^{(0)}$ be the input embedding of node v to the graph.

$$h_v^{(0)} = x_v, \text{ where } x_v \text{ is the node feature for node } v \quad (7.1)$$

Each GCN layer performs the following operation (Chen *et al.*, 2019):

$$h_v^{(k)} = f^{(k)} \left[W^{(k)} \cdot \frac{\sum_{u \in N(v)} h_u^{(k-1)}}{|N(v)|} + B^{(k)} \cdot h_v^{(k-1)} \right] \quad (7.2)$$

In the above equation, the layer takes in the $(k-1)^{th}$ input and performs a normalized

weighted mean of the previous layers' node neighbor embeddings, and then adds a bias term that just includes the node representation.

The proposed GC-module compresses feature embeddings obtained from a sentence transformer, and the adjacency matrix is obtained using METAR data to compute label co-occurrences independent of the co-occurrences in the imaging training set. To further enhance the predictive capabilities, an ensemble of GCN modules is used, with each being constructed using a different sentence embedding representation. The processed embeddings from the GCN are then fused using a weighted fusion module. The module consists of two main components:

1. Gating Network:

$$G_i = \sigma(W_i \cdot R_i) \quad (7.3)$$

2. Fusion Operation:

$$W_i = \frac{\exp(G_i)}{\sum_{j=1}^N \exp(G_j)} \quad (7.4)$$

$$F = \sum_{i=1}^N W_i \odot R_i \quad (7.5)$$

In these equations, R_i represents the input representation obtained from the i -th model, W_i represents the gating weights for the i -th representation, G_i represents the gated output for the i -th representation, and F represents the fused representation. Equation (7.3) calculates the gated output G_i by applying a set of transformations W_i to the input representation R_i using the sigmoid activation function σ . Equation (7.4) normalizes the gating weights G_i using the softmax function, ensuring they sum up to 1. Equation (7.5) fuses the representations by calculating the weighted sum, where W_i represents the normalized gating weights and R_i represents the input representations.

7.3.3.3 Sentence Transformer Module

The input embeddings to the graph representation learning module is obtained from the Sentence-BERT (S-BERT) model proposed in Reimers and Gurevych (2019) to get contextualized embeddings. The target variables of interest are represented as nodes in the above graph model. Phrases are created for the computation of the embeddings using a pre-trained model. The S-BERT model consists of stacked BERT encoding layers that consist of two repeating sub-layers: a multi-head self-attention mechanism followed by a fully-connected layer.

7.3.3.4 Multitask Supervised Learning

The representations obtained from the image encoding and ensemble-GCN module are then fused using a dot product to get the final logits. These logits are divided into the aforementioned 3 categories with 5 multilabel weather threats, 6 cloud covers, and 3 meteorological condition classes, respectively. The weather threats are learned using the multilabel version of the asymmetric loss Ben-Baruch *et al.* (2020). This loss function introduces different penalties for misclassifying positive and negative examples for each label. Given that some classes are rarer than others in the dataset, misclassifying a rare label should be penalized more heavily than misclassifying a more common label. That more common labels are to be given priority to be classified correctly is a legitimate viewpoint. Indeed, one might expect that the classifier is expected to classify commonly occurring labels more accurately. However, from the standpoint of empirical risk minimization of the model parameters, not weighting class occurrences either while sampling or in the loss function would result in over-fitting

the model to learning features of classes that occur more often. The binarized version of the loss is given by:

$$AL(p, y)_+ = (1 - p)_{+}^{\gamma} \log(p) \quad (7.6)$$

$$AL(p, y)_- = p_{-}^{\gamma} \log(1 - p) \quad (7.7)$$

In the above equations, γ_+ and γ_- are the weighting parameters for the penalty on rare positives and common negative labels. For the multilabel predictions, one can use the same loss by first using the sigmoid on the logits, and then aggregating the binary losses from each of the K labels.

To further regularize the model, an additional cost-matrix (Galdran *et al.*, 2020) for the cloud classification task is proposed. The motivation behind using the cost matrix is to provide more conservative cloud cover predictions. This would mean assigning an increasing cost to under-predicting the cloud cover level.

Model uncertainties are then obtained using methods similar to those used in the previous chapters. This study MC-Dropout (MCD), and the Concrete Dropout (CD). The former approach uses the dropout hyperparameter after each layer to randomly drop some activation to provide a model-averaging effect. The latter approximates a drop probability using a regularized loss formulation, leading to tighter uncertainty estimates.

7.4 Experimental Setup

7.4.1 GA-SA Dataset

The dataset comprises 178 labeled images, which is divided into 121 samples for training and 57 for the test set.

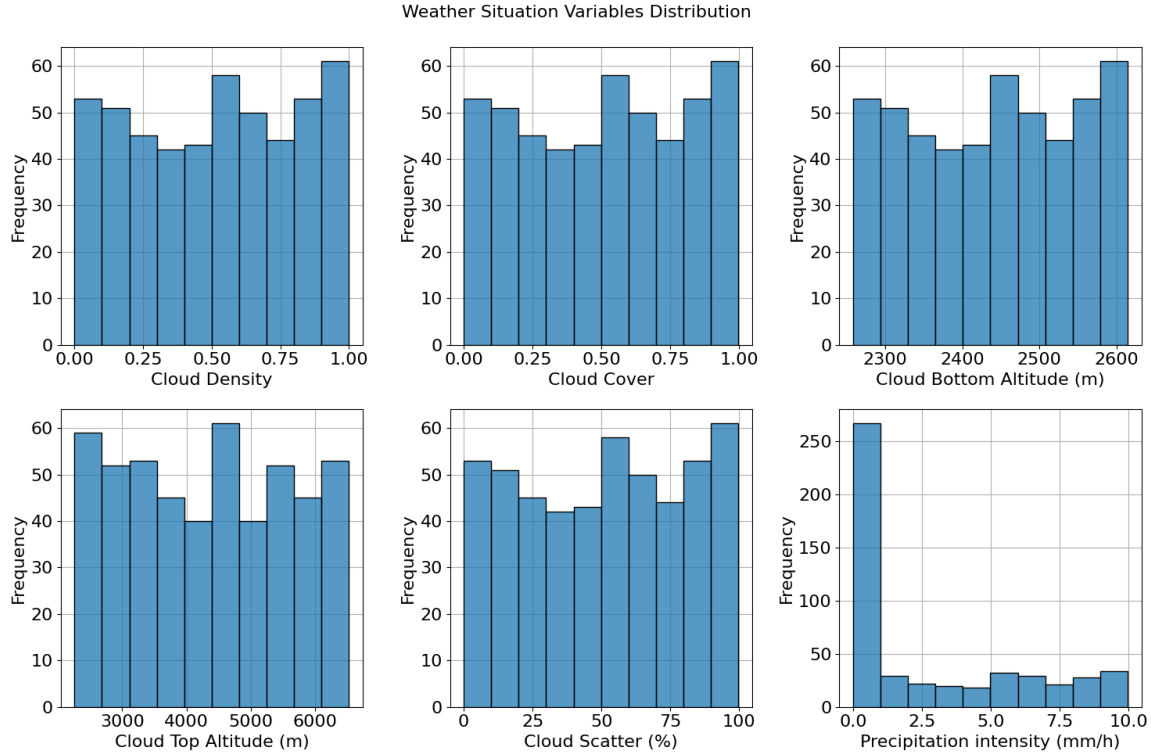


Figure 37. Distribution of weather parameters used in the generated dataset

Data is acquired from the state-of-the-art Microsoft Flight Simulator (MSFS). Compared to older simulators, this platform offers significant benefits in the areas of weather data and depiction, such as volumetric atmosphere representation including clouds, aerosol effects, and rain. The platform also provides real-world weather conditions using mesoscale weather models and METAR data, that would be useful for future works. The weather engine also accounts for atmospheric conditions like wind, terrain, and cloud formations, evolving them dynamically, making MSFS an attractive digital twin option for data acquisition. Images were captured from the Aspen Pitkin County airfield (KASE), utilizing a Cessna 152 airplane at fixed coordinates (39.253, 106.884) and an altitude of approximately 1500 ft AGL. This setup, by fixing the position, offers advantages for model training and data labeling. To introduce variability in weather data, scene parameters such as cloud top and bottom altitudes,

cloud density, cloud cover, cloud scatter, and precipitation level were randomized in the simulator. The images were then manually labeled. Figure 37 shows histograms of the various weather situation variables:

- Cloud Density: Varied between 0.0 and 1.0, with a relatively uniform distribution. The cloud density specifies the volumetric density of clouds, providing us with a variation in atmospheric lighting and cloud textures.
- Cloud Cover Percentage: Normalized values between 0 and 1, also uniformly distributed.
- Cloud Bottom Altitude: Ranging from 2250 m to 2600 m, with variable frequencies.
- Cloud Top Altitude: Ranging from 3000 m to 6000 m, showing a more varied distribution.
- Cloud Scatter: Percentage values from 0 to over 100%, with an uneven distribution.
- Precipitation Intensity: Measured in mm/h, most values are near 0, indicating low intensity, with few instances of higher intensities. This was done to ensure there is a good diversity in precipitation, and enough images without precipitation variables.

Images are collected out of the Aspen Pitkin County airfield (KASE), due to the complex terrain features surrounding it. The position and altitude of the aircraft are fixed throughout the time of recording, with position coordinates (39.253, 106.884) and an altitude of approximately 1500 ft AGL. This experimental setup of fixing the position of the aircraft provides two advantages. First, fixing the terrain features enables the model to learn the weather features on each image. Secondly, the fixed terrain features imply that the distances to points of interest in the scene are known and can be used as indicators for labeling the data in a more faithful manner to what the FAA standards require. The randomized weather parameters, shown in Figure 37 were then injected into the simulator for recording the images. The acquired images were then manually labeled. A few representative examples of the scene used for data



Figure 38. Representative images from the GA-Situational Awareness Dataset

collection are shown in Figure 38. There are several points of interest with known distances that can be acquired from map data which can be used to estimate ambient visibility for assigning MC head labels.

7.4.1.1 Weather Description using METAR and Phrase Embeddings

METAR data is collected from Boulder airport (KBDU) weather station, which is also present in the Rockies and presents similar terrain and weather features. 5 years of METAR data are collected and used to perform a series of data processing steps to obtain an adjacency matrix, as specified in Section 7.3. The METAR data is used to collect the frequencies of each label in the prediction, with the exception of TEC-PA, as that is a parameter that is only valid from the camera point-of-view. While the FG, BR, MIFG, and RA/SN categories can be directly extracted from the data, only the first layer of cloud data is considered for the cloud cover frequency computations. This is because the METAR data arranges cloud layers in ascending order of altitude. Typically, the first layer holds the utmost importance for VFR operations and aligns more closely with the collected image data. For instance, scattered clouds at the

same altitude as the aircraft have a greater impact on the probability of inadvertent IMC (Instrument Meteorological Conditions) entry compared to having broken clouds at higher altitudes. IMC, M-VMC, and VMC frequencies are computed using the METAR visibility data.

The next input to the Graph Convolutional module is the phrase embedding. Phrase embeddings are obtained from 3 pre-trained S-BERT models, with the non-ensembled version using a 768-dimensional MPNet Model Song *et al.* (2020). The ensemble model uses the MPNet and two MiniLM models with dimensions 768, 384, and 384, respectively Wang *et al.* (2020a).

7.4.2 Training and Evaluation Setup

The base setting for the ML-GCN model uses an embedding dimension of 384, and the total number of categories the model predicts is 14. A dropout probability of 0.2 is employed for the MC-Dropout experiments. For the ensemble model, a combination of sentence transformer embeddings from the MiniLM-L6, MiniLM-L12 and the MPNet models is used. The embedding dimensions are 384, 384, and 768, respectively.

During the training process, the Adam optimizer is used with a base learning rate (lr) set to 1e-5 and a maximum learning rate set to 5e-4. To efficiently schedule the learning rate, the OneCycleLR scheduler (Smith and Topin, 2018) is employed. This enables better convergence and improved generalization during the 300 epochs of training, with a batch size of 32.

The models that are used to compare against the proposed architecture are a baseline CNN model with a ResNet-18 encoder followed by a dense classifier layer, the Multilabel Supervised Contrastive Network (MulCon) (Dao *et al.*, 2021), and the

Multi-label Graph Convolutional Network (ML-GCN) (Chen *et al.*, 2019). All models use the ResNet-18 encoder as a common backbone, and pre-trained models are not used for the experiments.

7.4.3 Evaluation Metrics

For evaluating the model, the mean Average Precision score (mAP) and the F1 score for each head is computed. The F-1 score can be obtained as the harmonic mean of the precision and recall of the prediction:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \quad (7.8)$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad (7.9)$$

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (7.10)$$

Average Precision (AP) metric is computed by taking the weighted mean of precisions obtained at different confidence thresholds. The increase in recall from the previous threshold acts as the weight in this calculation. In essence, AP provides a single value that represents the overall performance of the model in terms of precision and recall.

$$AP = \sum_n (R_n - R_{n-1})P_n \quad (7.11)$$

In the above equation, R_n is the recall and P_n is the precision at the n^{th} confidence threshold. Micro-averaging is used to report both the F-1 score and the mAP score.

7.5 Results and Discussion

7.5.1 Model Evaluation

Table 17. Performance comparison across baseline model architectures

Method	Micro F1				mAP			
	WT	CC	MC	AVG	WT	CC	MC	AVG
Baseline	<u>84.57</u>	64.91	84.21	77.89	<u>74.61</u>	63.62	89.14	75.79
MulCon	74.95	37.5	82.4	64.95	40.66	46.91	72.9	53.5
ML-GCN	83.14	57.89	<u>87.71</u>	76.24	72.37	<u>67.43</u>	<u>92.31</u>	<u>77.37</u>
Ours	88.13	70.12	82.45	80.23	79.50	66.73	87.69	77.97
Ours-Ensemble	74.99	<u>68.22</u>	90.62	<u>77.94</u>	65.88	68.79	93.75	76.14

Table 17 shows the comparative benchmark result, which compares the performance of the proposed model against a baseline and two other recent approaches. The best-performing results are highlighted in bold, and the second-best are indicated with an underline. As discussed in Section 7.4, the baseline model comprises an encoder followed by a linear classifier. The other models that are benchmarked against are the MulCon Dao *et al.* (2021) and ML-GCN Chen *et al.* (2019) models. The raw performance of our model, the CS-sBERT-GCN, demonstrates significant improvements in mean F1 (mF1) and mAP scores compared to other baselines. Particularly, the cloud cover prediction excels in the CS-GCN model. The ML-GCN model, without any cost-sensitive learning, outperforms the proposed model in the MC head. The ensemble model consistently shows competitive performance with the other models. Notably, the proposed models outperform all other models. Another interesting result is that the baseline CNN model performs well in the weather threat

multilabel prediction head and is not the worst of the models. Surprisingly, the MulCon model performs significantly worse than the proposed models and the other baselines. It was expected that contrastive learning method proposed in MulCon would improve performance, but this was not observed in the results. For a deeper understanding of these results, future work will need to involve analyzing the features learned by these models.

To further investigate the effect of regularization, the base ML-GCN model is regularized with contrastive and cost-sensitive learning losses on the classification head. Table 18 shows the category-wise raw performance of the models. It is found that cost-sensitive learning with a conservative cost matrix most improves the performance over the ML-GCN baseline. As mentioned earlier, incorporation of a contrastive loss for cloud classification does not help with improving performance. This is unexpected because, on its own, contrastive loss with label information tends to pull dissimilar representations away from each other and pulls similar classes closer together in the latent space of the model. Contrastive learning is a promising direction of research, however, design of the architecture and training process to effectively take advantage of this regularizer needs to be explored in the future. Next, results for two dropout-based UQ approaches introduced in Section 7.3, MC-Dropout and ConcreteDropout are presented. The results in Table 19 show a drop in the variance when using ConcreteDropout. Future work should expand these results to incorporate a thorough benchmarking of UQ methods on the multi-label, multi-class classification problem.

Table 18. Impact of regularization on F-1 score

Method	WT	CC	MC	Avg
ML-GCN	83.14	57.89	87.71	76.24
Contrastive	75.49	55.2	86.63	72.44
Cost Sensitive	88.13	70.12	82.45	80.23

Table 19. F1 scores and variances for two dropout-based UQ methods

Method	WT F1	WT Var	CC F1	CC VAR	MET F1	MET VAR
MC-Dropout	85.71	1.87	61.64	0.957	82.19	1.44
ConcreteDropout	82.87	0.09	64.38	0.08	84.21	0.16

Finally, the sensitivity of the model to changes in the sentence embedding input phrase and the embedding model is shown in Table 20. It becomes evident that employing larger sentence embedding models to train the GC module yields significant benefits. Moreover, it is observed that the ensemble of all sBERT embedding models achieves higher mAP scores for cloud cover prediction and meteorological condition prediction. The ensemble also performs well in terms of F-1 scores for all heads, except the weather threat prediction head. However, the notably lower performance in the WT head for the embedding ensemble leads to an overall score lower than that of the MPNet model.

Table 20. Performance across sentence embedding models

Model	Micro F1				mAP			
	WT	CC	MC	AVG	WT	CC	MC	AVG
MiniLM-L6	<u>86.03</u>	61.40	89.47	78.96	<u>76.29</u>	62.71	<u>92.58</u>	77.19
MiniLM-L12	84.57	63.15	<u>89.47</u>	<u>79.06</u>	74.61	68.18	92.05	78.28
MPNet	88.13	70.12	82.45	80.23	79.50	<u>66.73</u>	87.69	<u>77.97</u>
Ensemble	74.99	<u>68.22</u>	90.62	77.94	65.88	68.79	93.75	76.14

7.5.2 Discussion and Limitations

The primary goal of this work is to showcase the potential of machine vision techniques to support pilot decision making. Adding other useful sources of data

such as a large database of METAR reports and sentence descriptors of weather phenomena can aid in improving predictive performance. A limitation is that data from flight simulators was used instead of real world images for training. While the simulator environment offers us high fidelity visuals, and an ability to exercise a great degree of control over the synthesis of the data, it does not exactly match real world images, and does not have the imaging imperfections seen in the real world. Another reason for this was due to the prohibitive cost and time of acquiring these images. Despite this limitation, this study highlights how such models can be used by pilots. Trained with real-world training data, the proposed model can output various threat categories at close to real-time. The only input that the model needs would be the image feed from the camera. The outputs from the model will be a vector of threats, as described earlier. The proposed detection model leaves a lot of potential for applications that improve pilot situational awareness. Since a description of threats is provided by the model and information about the aircraft state vector is already available, future work can focus on using these pieces of data to passively recommend trajectories to pilots, helping them comply with VFR regulations. For instance, this could be used to recommend evasive trajectories and alternate flight paths in case the pilot encounters rising terrain combined with clouds. Moreover, the incorporation of additional data sources to extend the proposed framework holds the potential to significantly improve pilot decision making. Examples of such sources include satellite imagery and terrain Digital Elevation Model (DEM) data. The inclusion of these sources can improve the accuracy of threat distance representation, enabling compliance with ceiling and horizontal distance limits for VFR flight. Additionally, conducting a feasibility study on incorporating depth information, with a focus on cloud detection using long-range depth estimation techniques from the existing literature,

would be beneficial. It is expected that challenges may arise in terms of experiment setup and data acquisition. The current dataset features a fixed terrain representation and varying weather conditions. Instead of modeling real-time weather within the simulator and synchronizing it with real-time satellite imagery, which would require substantial time for data collection and labeling, alternative and efficient methods for fusing satellite data can be explored.

7.6 Conclusion

In this chapter, a multi-label, multi-class threat detection model is proposed to improve GA pilot situational awareness. The contribution of this study is focused on preventing inadvertent IMC entry for GA pilots, and the proposed system provides a continuous monitoring tool that can serve as a part of a larger decision support system. This work is aimed to serve as a starting point for greater efforts in utilizing multiple modalities of data and cheap, commercially available sensing tools to provide a rich scene understanding model for GA pilots. This would eventually serve to augment NEXRAD data and METAR reports from airports, and not require GA pilots to mount expensive, bulky on-board weather radars. The contributions in this work include a deep learning model that combines METAR data, imaging data, and sentence embedding data to predict significant weather threats, cloud cover levels and visibility levels. Results indicate that the proposed model outperforms 3 state of the art models. While the proposed model showed significant improvements over this CNN baseline, the CNN model consistently placed second in the comparisons. It was also found that larger sentence embedding models showed better raw performance, with the ensemble of all sentence embedding models showing good performance on all categories but the weather threat prediction task. This study also observes of incorporating a conservative cost-sensitive loss to the training technique, which resulted in more conservative predictions for cloud classification, and overall improvements to performance. Future work in this area can focus on two main aspects. Firstly, uncertainty estimation can focus on robustness and enabling applications like data quality assessment and out-of-distribution detection. Secondly, the integration of additional data sources such as satellite imagery and terrain DEM data holds promise

for further enhancing system accuracy and compliance with VFR flight limits. These advancements will contribute to the ongoing development of an efficient and reliable GA decision support system.

Chapter 8

CONCLUSION

This dissertation presents an exploration of Bayesian Neural Network models in engineering applications, with a focus on the uncertainties produced by these models, predicting under constraints, the usage of multiple data modalities, and the performance of these neural network models when only limited labeled data is available. These issues remain important in the application of machine learning in today's industry. Therefore, each chapter in this document has contributed to advancing the understanding and capabilities of neural networks in addressing these problems. Chapter 3 introduced the concept of Bayesian Entropy Neural Networks (BENN), a framework that integrates known physics and knowledge as constraints into neural network outputs. This development, particularly relevant in scenarios with limited data, demonstrated the potential of BENN in applications such as structural analysis and microstructure generation. Chapter 4 presented the Bayesian Boundary-Aware Convolutional Network (B-BACN), a model that quantifies both epistemic and aleatoric uncertainties in crack detection. B-BACN's multi-task learning approach and the use of methods like MC-Dropout and Concrete Dropout have shown improvements in the precision and calibration of crack detection models. Chapter 5 explored the fusion of RGB and features from point-cloud data, leading to the development of a geometry-aware neural network for improved defect detection in gas pipelines. This approach demonstrated improvements in detection performance, illustrating the effectiveness of data fusion in pipeline monitoring. In Chapter 6, a novel semi-supervised learning approach for semantic segmentation in defect inspection

was introduced. This approach leveraged activation map interpolation and consistency regularization to achieve accurate defect segmentation with limited labeled data, addressing a significant challenge in industrial contexts. Chapter 7 proposed a decision support system for General Aviation (GA) pilots. This system combined imaging data, METAR reports, and sentence embeddings for a weather threat detection model, ultimately showcasing the potential of using imaging systems in conjunction with other relevant data sources to improve situational awareness for pilots.

This dissertation has demonstrated the challenges associated with accurately modeling prediction confidence. From the Bayesian point of view, the word "accurate" for prediction confidence becomes rather nebulous. Subjectivity enters the picture quite early in the modeling phase, given that priors are set on parameters and model choice implicitly restricts the class of functions that can be learned. Approximate inference techniques also have various hyperparameters that need to be tuned based on the application in consideration. Proponents of alternative statistical inference methods, such as those based on Conformal Prediction, have argued that Bayesian methods are prone to misspecification. The risk with Bayesian models is that if these priors are misspecified – either due to incorrect assumptions, lack of prior knowledge, biases or the difficulty in directly specifying weight parameter priors for expressing beliefs about the posterior predictive – the resulting inference can be misleading. This risk is compounded in complex models where the true data-generating process is unknown or difficult to approximate accurately. An avenue of research is the integration of principles from conformal prediction with Bayesian models. This hybrid approach could potentially combine the strengths of both methodologies in different steps of the machine learning and decision support pipeline.

The results in Chapter 4 also show that the relationship between the amount

of training data and model uncertainty for image segmentation problems is not monotonic – Developing a novel uncertainty quantification technique that better aligns the uncertainty to be more sensitive to the distance from training image samples is an important problem for building models to detect distribution shifts better. While work has begun in this direction, and the Bayesian Entropy Method offers an attractive option to directly express beliefs about the decay in confidence of the model predictions, principled alternatives can be developed. Key to this might be working on making the reparameterization step more expressive to bring in more nuanced formulations of the variance. Further, it is worth noting that methods such as Bayes By Backprop and Deep Ensembles have higher training time requirements due to the increased number of parameters or models. Conversely, MC Dropout has higher prediction time requirements for computing sampled uncertainty estimates. Therefore, future work should focus on achieving uncertainty estimates efficiently. To accomplish this, foundational ideas from (Liu *et al.*, 2020a) and (Anirudh and Thiagarajan, 2021) can be borrowed, along with recent work on noise propagation for uncertainty estimates, as presented in (Postels *et al.*, 2019).

Uncertainty estimation for multi-label classification is a crucial challenge that has seen specialized methods developed in the literature. Current work only applies dropout-based Bayesian Deep Learning techniques to multi-label prediction uncertainty. To address this, future work should include thorough benchmarking of uncertainty quantification (UQ) approaches, encompassing both Bayesian and non-Bayesian methods within this problem domain. This research will contribute to developing a robust UQ method for multi-label, multi-class settings, enabling various applications such as assessing input data quality, robustness to attacks, and out-of-distribution (OOD) detection.

From the point of view of applications, Chapter 7 presents a research study that offers several potential directions for further investigation aimed at developing scalable solutions for GA decision support systems. The focus of the study in this dissertation was limited to utilizing camera sensors to gather comprehensive scene information, along with METAR data. To enhance the developed system, future work can concentrate on the utilization of data fusion techniques to provide more comprehensive decision support. The incorporation of additional data sources into the currently proposed framework holds the potential to enhance system performance significantly. Examples of such sources include satellite imagery and terrain Digital Elevation Model (DEM) data. The inclusion of these sources can improve the accuracy of threat distance representation, enabling compliance with ceiling and horizontal distance limits for VFR flight. Additionally, conducting a feasibility study on incorporating depth information, focusing on cloud detection using long-range depth estimation techniques from the existing literature, would be beneficial. It is important to note that challenges may arise in terms of experiment setup and data acquisition.

REFERENCES

- Abdar, M., F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, X. Cao, A. Khosravi, A. Khosravi, U. R. Acharya, U. R. Acharya, V. Makarenkov, S. Nahavandi and S. Nahavandi, “A review of uncertainty quantification in deep learning: Techniques, applications and challenges”, arXiv: Learning (2021).
- Alofi, A., A. A. Alghamdi, R. Alahmadi, N. Aljuaid and M. Hemalatha, “A review of data fusion techniques”, International Journal of Computer Applications **167**, 37–41 (2017).
- Alzuhiri, M., K. Farrag, E. Lever and Y. Deng, “An electronically stabilized multi-color multi-ring structured light sensor for gas pipelines internal surface inspection”, IEEE Sensors Journal **21**, 17, 19416–19426 (2021).
- Anirudh, R. and J. Thiagarajan, “ δ -uq: Accurate uncertainty quantification via anchor marginalization”, null (2021).
- Ayenu-Prah, A. Y., N. Attoh-Okine, N. O. Attoh-Okine and N. O. Attoh-Okine, “Evaluating pavement cracks with bidimensional empirical mode decomposition”, EURASIP Journal on Advances in Signal Processing (2008).
- Ben-Baruch, E., T. Ridnik, N. Zamir, A. Noy, I. Friedman, M. Protter, M. Protter and L. Zelnik-Manor, “Asymmetric loss for multi-label classification”, arXiv: Computer Vision and Pattern Recognition (2020).
- Berman, M., A. R. Triki and M. B. Blaschko, “The lovasz-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks”, URL <https://arxiv.org/abs/1705.08790> (2017).
- Berman, M., A. R. Triki and M. B. Blaschko, “The lovasz-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks”, 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (2018).
- Besl, P. J. and R. C. Jain, “Invariant surface characteristics for 3d object recognition in range images”, Computer Vision, Graphics, and Image Processing **33**, 1, 33–80, URL <https://www.sciencedirect.com/science/article/pii/0734189X86902203> (1986).
- Bishop, C. M. and N. M. Nasrabadi, *Pattern recognition and machine learning*, vol. 4 (Springer, 2006), URL <https://dl.acm.org/doi/10.5555/1162264>.

- Bivalkar, M., S. Agarwal and D. Singh, “Development of an efficient approach for detection and measurement of crack length in ceramic tile manufacturing using millimeter-wave imaging”, *NDT & E International* **129**, 102656 (2022).
- Blei, D. M., A. Kucukelbir and J. D. McAuliffe, “Variational inference: A review for statisticians”, *Journal of the American statistical Association* **112**, 518, 859–877 (2017).
- Blundell, C., J. Cornebise, K. Kavukcuoglu and D. Wierstra, “Weight uncertainty in neural network”, in “International conference on machine learning”, pp. 1613–1622 (PMLR, 2015).
- Bottou, L. *et al.*, “Stochastic gradient learning in neural networks”, *Proceedings of Neuro-Nimes* **91**, 8, 12 (1991).
- Brach, K., B. Sick and O. Dürr, “Single shot mc dropout approximation”, arXiv preprint (2020).
- Bradski, G., “The opencv library.”, *Dr. Dobb’s Journal: Software Tools for the Professional Programmer* **25**, 11, 120–123 (2000).
- Cao, Y., W. Tan, Z. Wu, Z. Wu and Z. Wu, “Aircraft icing: an ongoing threat to aviation safety”, *Aerospace Science and Technology* (2018).
- Chan, T. F., L. A. Vese and L. A. Vese, “Active contours without edges”, *IEEE Transactions on Image Processing* (2001).
- Chang, C.-W., T.-M. Chao, J.-T. Horng, C.-F. Lu and R.-H. Yeh, “Development pattern recognition model for the classification of circuit probe wafer maps on semiconductors”, *IEEE Transactions on Components, Packaging and Manufacturing Technology* **2**, 12, 2089–2097 (2012).
- Chase, H. S., “Fundamental forms of surfaces and the gauss-bonnet theorem”, (2015).
- Chen, J., C. van Eeden and J. Zidek, “Uncertainty and the conditional variance”, *Statistics and Probability Letters* **80**, 23, 1764–1770, URL <https://www.sciencedirect.com/science/article/pii/S0167715210002154> (2010).
- Chen, W., Y. Gao, L. Gao and X. Li, “A New Ensemble Approach based on Deep Convolutional Neural Networks for Steel Surface Defect classification”, in “Procedia CIRP”, (2018).
- Chen, X., Y. Lian, L. Jiao, H. Wang, Y. Gao and S. Lingling, “Supervised edge attention network for accurate image instance segmentation”, in “European Conference on Computer Vision”, pp. 617–631 (Springer, 2020).

- Chen, X., Y. Yuan, G. Zeng and J. Wang, “Semi-supervised semantic segmentation with cross pseudo supervision”, *Computer Vision and Pattern Recognition* (2021).
- Chen, Z.-M., X.-S. Wei, P. Wang, P. Wang, P. Wang, P. Wang, P. Wang and Y. Guo, “Multi-label image recognition with graph convolutional networks”, *Computer Vision and Pattern Recognition* (2019).
- Cheng, W. and Y. Zhou, “Automatic pavement crack detection based on hierarchical feature augmentation”, *International Conference on Artificial Intelligence and Information Systems* (2021).
- Cheng, X. and J. Yu, “RetinaNet with Difference Channel Attention and Adaptively Spatial Feature Fusion for Steel Surface Defect Detection”, *IEEE Transactions on Instrumentation and Measurement* (2021).
- Ciccio, C. D., H. van der Aa, C. Cabanillas, J. Mendling and J. Prescher, “Detecting flight trajectory anomalies and predicting diversions in freight transportation”, *Entwicklungsmethoden für Informationssysteme und deren Anwendung: Fachtagung* (2016).
- Cordts, M., M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth and B. Schiele, “The cityscapes dataset for semantic urban scene understanding”, in “*Proceedings of the IEEE conference on computer vision and pattern recognition*”, pp. 3213–3223 (2016).
- Dao, S. D., E. Zhao, D. Phung and J. Cai, “Multi-label image classification with contrastive learning”, *arXiv.org* (2021).
- de Mathelin, A., F. Deheeger, M. Mougeot and N. Vayatis, “Maximum weight entropy”, *arXiv preprint arXiv:2309.15704* (2023).
- Deng, J., Y. Lu and V. C.-S. Lee, “Imaging-based crack detection on concrete surfaces using you only look once network”, *Structural Health Monitoring* **20**, 2, 484–499 (2021).
- Di, H., X. Ke, Z. Peng and Z. Dongdong, “Surface defect classification of steels with a new semi-supervised learning method”, *Optics and Lasers in Engineering* (2019).
- Din, S. U., J. Shao, J. Kumar, W. Ali, J. Liu and Y. Ye, “Online reliable semi-supervised learning on evolving data streams”, *Information Sciences* **525**, 153–171 (2020).
- Dong, H., K. Song, Y. He, J. Xu, Y. Yan and Q. Meng, “Pga-net: Pyramid feature fusion and global context attention network for automated surface defect detection”, *IEEE Transactions on Industrial Informatics* (2019).

- Dong, H., K. Song, Y. He, J. Xu, Y. Yan and Q. Meng, “PGA-Net: Pyramid Feature Fusion and Global Context Attention Network for Automated Surface Defect Detection”, *IEEE Transactions on Industrial Informatics* (2020).
- Fan, J., B.-B. Gao, H. Jin and L. Jiang, “Ucc: Uncertainty guided cross-head co-training for semi-supervised semantic segmentation”, *Computer Vision and Pattern Recognition* (2022).
- Fan, X., X. Zhang and X. B. Yu, “Uncertainty quantification of a deep learning model for failure rate prediction of water distribution networks”, *Reliability Engineering and System Safety* (2023).
- Fan, Z., Y. Wu, J. Lu and W. Li, “Automatic pavement crack detection based on structured prediction with the convolutional neural network”, *arXiv preprint* (2018).
- Fernández-Moreno, M., B. Lei, E. A. Holm, P. Mesejo and R. Moreno, “Exploring the trade-off between performance and annotation complexity in semantic segmentation”, *Engineering applications of artificial intelligence* (2023).
- Frank, D., J. Chhor and R. Schmitt, “Stereo-vision for autonomous industrial inspection robots”, in “2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)”, pp. 2555–2561 (IEEE, 2017).
- French, G., T. Aila, S. Laine, M. Mackiewicz and G. Finlayson, “Consistency regularization and CutMix for semi-supervised semantic segmentation”, (2019).
- Gal, Y. and Z. Ghahramani, “Dropout as a bayesian approximation: Representing model uncertainty in deep learning”, in “international conference on machine learning”, pp. 1050–1059 (PMLR, 2016), URL <https://proceedings.mlr.press/v48/gal16.html>.
- Gal, Y., J. Hron and A. Kendall, “Concrete dropout”, in “Advances in Neural Information Processing Systems”, edited by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan and R. Garnett, vol. 30, p. 0 (Curran Associates, Inc., 2017).
- Galdran, A., J. Dolz, H. Chakor, H. Lombaert and I. B. Ayed, “Cost-sensitive regularization for diabetic retinopathy grading from eye fundus images”, (2020).
- Gao, J., P. Li, Z. Chen and J. Zhang, “A survey on deep learning for multimodal data fusion”, *Neural Computation* **32**, 829–864 (2020).
- Gao, Y., Y. Jiao and Y. Liu, “Ultra-efficient reconstruction of 3d microstructure and distribution of properties of random heterogeneous materials containing multiple phases”, *Acta Materialia* **204**, 116526 (2021).

- Giffin, A., A. Caticha and A. Caticha, “Updating probabilities with data and moments”, arXiv: Data Analysis, Statistics and Probability (2007).
- Girshick, R., “Fast r-cnn”, IEEE International Conference on Computer Vision (2015).
- Goldberg, Y., “A primer on neural network models for natural language processing”, Journal of Artificial Intelligence Research **57**, 345–420 (2016).
- Groenendijk, R., S. Karaoglu, T. Gevers and T. Mensink, “Multi-loss weighting with coefficient of variations”, (2020).
- Gu, J., J. Gu, Z. Wang, Z. Wang, J. Kuen, J. Kuen, L. Ma, L. Ma, A. Shahroudy, A. Shahroudy, A. Shahroudy, B. Shuai, B. Shuai, T. Liu, T. Liu, X. Wang, X. Wang, G. Wang, G. Wang, J. Cai, J. Cai, T. Chen and T. Chen, “Recent advances in convolutional neural networks”, Pattern Recognition (2018).
- Guo, C., G. Pleiss, Y. Sun and K. Q. Weinberger, “On calibration of modern neural networks”, URL <https://arxiv.org/abs/1706.04599> (2017).
- Guo, J.-M., H. Markoni and J.-D. Lee, “Barnet: Boundary aware refinement network for crack detection”, IEEE Transactions on Intelligent Transportation Systems (2021a).
- Guo, X., Y. Chen, Q. Si and W. Yuansheng, “Evolution mechanism on the unsafe behavioural risks of general aviation pilots”, The Engineering Economics (2021b).
- Gupta, S., R. Girshick, P. Arbeláez and J. Malik, “Learning rich features from rgb-d images for object detection and segmentation”, in “European conference on computer vision”, pp. 345–360 (Springer, 2014).
- Gutierrez, P., M. Luschkova, A. Cordier, M. Shukor, M. Schappert and T. Dahmen, “Synthetic training data generation for deep learning based quality inspection”, in “Proceedings of SPIE - The International Society for Optical Engineering”, vol. 11794 (SPIE, 2021).
- Gyimah, N. K., A. Girma, M. N. Mahmoud, S. Nateghi, A. Homaifar and D. Opoku, “A robust completed local binary pattern (rcbp) for surface defect detection”, in “2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)”, pp. 1927–1934 (IEEE, 2021).
- Haciefendioglu, K. and H. B. Basaga, “Concrete road crack detection using deep learning-based faster r-cnn method”, Iranian Journal of Science and Technology-Transactions of Civil Engineering (2021).
- Hartley, R. and A. Zisserman, *Multiple view geometry in computer vision* (Cambridge university press, 2003).

- Hawari, A., M. Alamin, F. Alkadour, M. Elmasry and T. Zayed, “Automated defect detection tool for closed circuit television (cctv) inspected sewer pipelines”, *Automation in Construction* **89**, 99–109 (2018).
- He, K., X. Zhang, S. Ren and J. Sun, “Deep residual learning for image recognition”, *Computer Vision and Pattern Recognition* (2016a).
- He, K., X. Zhang, S. Ren and J. Sun, “Deep residual learning for image recognition”, in “*Proceedings of the IEEE conference on computer vision and pattern recognition*”, pp. 770–778 (2016b).
- He, Y., K. Song, Q. Meng and Y. Yan, “An end-to-end steel surface defect detection approach via fusing multiple hierarchical features”, *IEEE Transactions on Instrumentation and Measurement* **69**, 4, 1493–1504 (2019).
- He, Z. and Q. Liu, “Deep Regression Neural Network for Industrial Surface Defect Detection”, *IEEE Access* (2020).
- Helifa, B., A. Oulhadj, A. Benbelghit, I. K. Lefkaier, F. Boubenider and D. Boutasouna, “Detection and measurement of surface cracks in ferromagnetic materials using eddy current testing”, *NDT and E International* (2006).
- Hosseini-Asl, E., J. M. Zurada and O. Nasraoui, “Deep learning of part-based representation of data using sparse autoencoders with nonnegativity constraints”, *IEEE transactions on neural networks and learning systems* **27**, 12, 2486–2498 (2015).
- Hu, L., J. Li, X. Peng, J. Xiao, B. Zhan, C. Zu, X. Wu, J. Zhou and Y. Wang, “Semi-supervised npc segmentation with uncertainty and attention guided consistency”, *Knowledge-Based Systems* **239**, 108021, URL <https://www.sciencedirect.com/science/article/pii/S0950705121011205> (2022).
- Huang, J., J. Huang, Y. Pang, Y. Pang, Y. Liu, Y. Liu, H. Yan and H. Yan, “Posterior regularized bayesian neural network incorporating soft and hard knowledge constraints”, *Cornell University - arXiv* (2022).
- Huang, Y., C. Qiu, Y. Guo, Y. Guo, X. Wang and K. Yuan, “Surface defect saliency of magnetic tile”, *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)* (2018).
- Huynh, P., R. Ross, A. Martchenko and J. Devlin, “Anomaly inspection in sewer pipes using stereo vision”, in “*2015 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*”, pp. 60–64 (IEEE, 2015).
- Huynh, P., R. Ross, A. Martchenko and J. Devlin, “3d anomaly inspection system for sewer pipes using stereo vision and novel image processing”, in “*2016 IEEE*

- 11th Conference on Industrial Electronics and Applications (ICIEA)", pp. 988–993 (IEEE, 2016).
- Iqbal, S., T. M. Khan, S. S. Naqvi and G. Holmes, "Mlr-net: A multi-layer residual convolutional neural network for leather defect segmentation", *Engineering applications of artificial intelligence* (2023).
- Iyer, S., S. K. Sinha, M. K. Pedrick and B. R. Tittmann, "Evaluation of ultrasonic inspection and imaging systems for concrete pipes", (2012).
- Jeong, S., K. You, D. Suk and D. Seok, "Hazardous flight region prediction for a small uav operated in an urban area using a deep neural network", *Aerospace Science and Technology* (2021).
- Johnson, C. M. and D. A. Wiegmann, "Vfr into imc: Using simulation to improve weather-related decision-making", *The International Journal of Aviation Psychology* (2015).
- Johnson, I., G. Whitehurst, V. N. Risukhin, L. J. Brown, W. G. Rantz, T. K. Ferris, T. Roady, C. Rodriguez-Paras, K. G. Tippey and M. J. Futrell, "Pegasas: Weather technology in the cockpit", null (2017).
- Kato, S., T. Hino, S. Kume and H. Nobuhara, "Crack detection from weld bend test images using r-cnn", *International Conference on P2P, Parallel, Grid, Cloud and Internet Computing* (2022).
- Kavukcuoglu, K., P. Sermanet, Y.-L. Boureau, K. Gregor, M. Mathieu, Y. Cun *et al.*, "Learning convolutional feature hierarchies for visual recognition", *Advances in neural information processing systems* **23** (2010).
- Kelly, D. and M. Efthymiou, "An analysis of human factors in fifty controlled flight into terrain aviation accidents from 2007 to 2017", *Journal of Safety Research* (2019).
- Kendall, A. and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?", *CoRR* **abs/1703.04977** (2017).
- Khan, A., A. Sohail, U. Zahoora and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks", *Artificial intelligence review* **53**, 5455–5516 (2020).
- Kirschke, K. R., S. A. Velinsky and S. A. Velinsky, "Histogram-based approach for automated pavement-crack sensing", *Journal of Transportation Engineering-asce* (1992).

- Kong, L., X. Peng, Y. Chen, P. Wang and M. Xu, “Multi-sensor measurement and data fusion technology for manufacturing process monitoring: a literature review”, *International Journal of Extreme Manufacturing* **2**, 022001, URL <https://iopscience.iop.org/article/10.1088/2631-7990/ab7ae6https://iopscience.iop.org/article/10.1088/2631-7990/ab7ae6/meta> (2020).
- Krähenbühl, P. and V. Koltun, “Efficient inference in fully connected crfs with gaussian edge potentials”, *Advances in neural information processing systems* **24** (2011).
- Kramer, L., R. Bailey and L. Prinzel, “Crew decision-making using fused s/evs 1 running head: Crew decision-making used fused s/evs commercial flight crew decision-making during low-visibility approach operations using fused synthetic / enhanced vision systems”, null (2009).
- Krizhevsky, A., A. Krizhevsky, I. Sutskever, I. Sutskever, G. E. Hinton and G. E. Hinton, “Imagenet classification with deep convolutional neural networks”, *Communications of The ACM* (2017).
- LeCun, Y., Y. Bengio *et al.*, “Convolutional networks for images, speech, and time series”, *The handbook of brain theory and neural networks* **3361**, 10, 1995 (1995).
- Lee, D., S. Yoon, J. Park, S. Eum and H. Cho, “Demonstration of model-assisted probability of detection framework for ultrasonic inspection of cracks in compressor blades”, *NDT & E International* **128**, 102618 (2022).
- Lee, D.-H., “Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks”, *ICML 2013 Workshop: Challenges in Representation Learning* (2013).
- Lee, J.-G., S. Jun, Y.-W. Cho, H. Lee, G. B. Kim, J. B. Seo and N. Kim, “Deep learning in medical imaging: general overview”, *Korean journal of radiology* **18**, 4, 570–584 (2017).
- Li, J., Z. Su, J. Geng and Y. Yin, “Real-time Detection of Steel Strip Surface Defects Based on Improved YOLO Detection Network”, *IFAC-PapersOnLine* (2018).
- Li, Y., Z. Han, H. Xu, L. Liu, X. Li and K. Zhang, “YOLOv3-lite: A lightweight crack detection network for aircraft structure based on depthwise separable convolutions”, *Applied Sciences (Switzerland)* (2019).
- Lin, T.-Y., P. Goyal, R. Girshick, K. He and P. Dollár, “Focal loss for dense object detection”, (2018).
- Lin, T.-Y., T.-Y. Lin, P. Dollár, P. Dollar, R. Girshick, R. Girshick, K. He, K. He, B. Hariharan, B. Hariharan, S. Belongie and S. Belongie, “Feature pyramid networks for object detection”, *Computer Vision and Pattern Recognition* (2017).

- Ling, Z., A. Zhang, D. Ma, Y. Shi and H. Wen, “Deep siamese semantic segmentation network for pcb welding defect detection”, *IEEE Transactions on Instrumentation and Measurement* **71**, 1–11 (2022).
- Liu, C., L.-C. Chen, F. Schroff, H. Adam, W. Hua, A. L. Yuille and L. Fei-Fei, “Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation”, in “Proceedings of the IEEE/CVF conference on computer vision and pattern recognition”, pp. 82–92 (2019a).
- Liu, J. Z., Z. Lin, S. Padhy, D. Tran, T. Bedrax-Weiss and B. Lakshminarayanan, “Simple and principled uncertainty estimation with deterministic deep learning via distance awareness”, (2020a).
- Liu, M., Y. Liu, H. Hu and L. Nie, “Genetic algorithm and mathematical morphology based binarization method for strip steel defect image with non-uniform illumination”, *Journal of Visual Communication and Image Representation* (2016).
- Liu, S., S. Zhi, E. Johns and A. Davison, “Bootstrapping semantic segmentation with regional contrast”, *International Conference on Learning Representations* (2021).
- Liu, T. and Z. He, “Tas2-net: Triple-attention semantic segmentation network for small surface defect detection”, *IEEE Transactions on Instrumentation and Measurement* **71**, 1–12 (2022).
- Liu, Y., K. Xu and J. Xu, “An improved MB-LBP defect recognition approach for the surface of steel plates”, *Applied Sciences (Switzerland)* (2019b).
- Liu, Y., J. Yao, X. Lu, R. Xie and L. Li, “Deepcrack: A deep hierarchical feature learning architecture for crack segmentation”, *Neurocomputing* **338**, 139–153, URL <https://www.sciencedirect.com/science/article/pii/S0925231219300566> (2019c).
- Liu, Y., Y. Yuan, C. Balta and J. Liu, “A light-weight deep-learning model with multi-scale features for steel surface defect classification”, *Materials* (2020b).
- Liu, Z., H. Ukida, P. Ramuhalli and K. Niel, “Integrated imaging and vision techniques for industrial inspection”, *Advances in Computer Vision and Pattern Recognition* (2015).
- Long, J., E. Shelhamer and T. Darrell, “Fully convolutional networks for semantic segmentation”, *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2017).
- Luo, W., Y. Li, R. Urtasun and R. Zemel, “Understanding the effective receptive field in deep convolutional neural networks”, *Advances in neural information processing systems* **29** (2016).

- Lv, Z., Y. Li, S. Qian and L. Wu, “Online surface defect segmentation on aluminum strip production line using a lightweight and efficient model”, *Engineering applications of artificial intelligence* (2023).
- Madhavan, P. and F. C. Lacson, “Psychological factors affecting pilots’ decisions to navigate in deteriorating weather”, *North American Journal of Psychology* (2006).
- Majumdar, N., K. Marais and A. Rao, “ANALYSIS OF GENERAL AVIATION FIXED-WING AIRCRAFT ACCIDENTS INVOLVING INFLIGHT LOSS OF CONTROL USING a STATE-BASED APPROACH”, *Aviation* **25**, 4, 283–294, URL <https://doi.org/10.3846/aviation.2021.15837> (2021).
- Malassiotis, S. and M. Srinivasan, “Stereo vision system for precision dimensional inspection of 3d holes”, *Machine Vision and Applications* **15**, 101–113 (2003).
- Mao, Y., J. Chen, P. Ping, P. Ping, P. Ping and C. Hao, “Crack detection with multi-task enhanced faster r-cnn model”, *International Conference on Big Data Computing Service and Applications* (2020).
- Mao-de, Y., B. Shaobo, X. Kun and H. Yuyao, “Pavement crack detection and analysis for high-grade highway”, *International Conference on Electronic Measurement and Instruments* (2007).
- McFarland, J. and E. C. DeCarlo, “A monte carlo framework for probabilistic analysis and variance decomposition with distribution parameter uncertainty”, *Reliability Engineering and System Safety* (2020).
- Miyato, T., S. I. Maeda, M. Koyama and S. Ishii, “Virtual Adversarial Training: A Regularization Method for Supervised and Semi-Supervised Learning”, *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019).
- Moradi, R., S. Cofre-Martel, E. L. Droguett, M. Modarres and K. M. Groth, “Integration of deep learning and bayesian networks for condition and operation risk monitoring of complex engineering systems”, *Reliability Engineering & System Safety* (2022).
- Naeni, M. P., G. F. Cooper and M. Hauskrecht, “Obtaining well calibrated probabilities using bayesian binning”, in “Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence”, AAAI’15, p. 2901–2907 (AAAI Press, 2015).
- Nakamura, N., K. Ashida, T. Takishita, H. Ogi and M. Hirao, “Inspection of stress corrosion cracking in welded stainless steel pipe using point-focusing electromagnetic-acoustic transducer”, *Ndt & E International* **83**, 88–93 (2016).
- Nasr, D., W. Slika and G. Saad, “Uncertainty quantification for structural health monitoring applications”, *Smart Structures and Systems* **22**, 399 (2018).

- Opromolla, R. and G. Fasano, “Visual-based obstacle detection and tracking, and conflict detection for small uas sense and avoid”, *Aerospace Science and Technology* (2021).
- Ouali, Y., C. Hudelot and M. Tami, “Semi-supervised semantic segmentation with cross-consistency training”, *Computer Vision and Pattern Recognition* (2020).
- Painter, J., W. Kelly, W. E. Kelly, J. Trang, K. Lee, K. Lee, P. Branham, J. Crump, D. T. Ward, D. Ward, D. Ward, D. Ward, K. Krishnamurthy, K. Krishnamurthy, K. Krishnamurthy, K. Krishnamurthy, D. Woo, W. Alcorn, A. Robbins and R.-J. Yu, “Decision support for the general aviation pilot”, 1997 IEEE International Conference on Systems, Man, and Cybernetics. *Computational Cybernetics and Simulation* (1997).
- Pang, Y., X. Zhao, J. Hu, H. Yan and Y. Liu, “Bayesian spatio-temporal graph transformer network (b-star) for multi-aircraft trajectory prediction”, *Knowledge-Based Systems* p. 108998 (2022).
- Pang, Y., X. Zhao, H. Yan and Y. Liu, “Data-driven trajectory prediction with weather uncertainties: A bayesian deep learning approach”, *Transportation Research Part C: Emerging Technologies* **130**, 103326 (2021).
- Pin, W., P. Wang, E. Fan, E. Fan, P. Wang and P. Wang, “Comparative analysis of image classification algorithms based on traditional machine learning and deep learning”, *Pattern Recognition Letters* (2020).
- Platt, J., J. Platt, A. H. Barr and A. H. Barr, “Constrained differential optimization”, *Neural Information Processing Systems* (1987).
- Postels, J., F. Ferroni, H. Coskun, N. Navab and F. Tombari, “Sampling-free epistemic uncertainty estimation using approximated variance propagation”, in “Proceedings of the IEEE/CVF International Conference on Computer Vision”, pp. 2931–2940 (2019).
- Pyle, R. J., R. R. Hughes, A. A. S. Ali and P. D. Wilcox, “Uncertainty quantification for deep learning in ultrasonic crack characterization.”, *IEEE Transactions on Ultrasonics Ferroelectrics and Frequency Control* (2022).
- Qiao, S., W. Shen, Z. Zhang, B. Wang and A. L. Yuille, “Deep co-training for semi-supervised image recognition”, *European Conference on Computer Vision* (2018).
- Raissi, M., M. Raissi, P. Perdikaris, P. Perdikaris, G. Karniadakis and G. E. Karniadakis, “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations”, *Journal of Computational Physics* (2019).

- Ramasamy, S., R. Sabatini, A. Gardi and J. Liu, “Lidar obstacle warning and avoidance system for unmanned aerial vehicle sense-and-avoid”, *Aerospace Science and Technology* (2016).
- Rasmussen, C. B., K. Kirk and T. B. Moeslund, “The challenge of data annotation in deep learning—a case study on whole plant corn silage”, *Sensors* **22**, 4, 1596 (2022).
- Rathod, J. M. and H. Salehi, “Vision system with deep learning classifiers for automatic quality inspection”, in “Proceedings of SPIE - The International Society for Optical Engineering”, vol. 11400 (SPIE, 2020).
- Redmon, J., S. K. Divvala, R. Girshick and A. Farhadi, “You only look once: Unified, real-time object detection”, *Computer Vision and Pattern Recognition* (2016).
- Reimers, N. and I. Gurevych, “Sentence-bert: Sentence embeddings using siamese bert-networks”, *arXiv: Computation and Language* (2019).
- Ren, S., K. He, R. Girshick and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks”, *arXiv: Computer Vision and Pattern Recognition* (2015).
- Ronneberger, O., P. Fischer and T. Brox, “U-net: Convolutional networks for biomedical image segmentation”, *arXiv: Computer Vision and Pattern Recognition* (2015).
- Rumelhart, D. E., G. E. Hinton and R. J. Williams, “Learning representations by back-propagating errors”, *nature* **323**, 6088, 533–536 (1986).
- Saiz, F. A., I. Serrano, I. Barandiaran and J. R. Sanchez, “A Robust and Fast Deep Learning-Based Method for Defect Classification in Steel Surfaces”, in “9th International Conference on Intelligent Systems 2018: Theory, Research and Innovation in Applications, IS 2018 - Proceedings”, (2018).
- Sajedi, S. O. and X. Liang, “Uncertainty-assisted deep vision structural health monitoring”, *Computer-aided Civil and Infrastructure Engineering* (2020).
- Sankararaman, S. and S. Mahadevan, “Uncertainty quantification in structural health monitoring”, *AIAA* (2009).
- Seites-Rundlett, W., M. Z. Bashar, C. Torres-Machi and R. B. Corotis, “Combined evidence model to enhance pavement condition prediction from highly uncertain sensor data”, *Reliability Engineering & System Safety* (2021).
- Shafeek, H., E. Gadelmawla, A. Abdel-Shafy and I. Elewa, “Automatic inspection of gas pipeline welding defects using an expert vision system”, *NDT & E International* **37**, 4, 301–307 (2004).

- Shi, Y., L. Cui, Z. Qi, F. Meng and Z. Chen, “Automatic road crack detection using random structured forests”, *IEEE Transactions on Intelligent Transportation Systems* (2016).
- Shi, Y., C. Zhang, R. Li, M. Cai and G. Jia, “Theory and application of magnetic flux leakage pipeline detection”, (2015).
- Shi, Y., C. Zu, P. Yang, S. Tan, H. Ren, X. Wu, J. Zhou and Y. Wang, “Uncertainty-weighted and relation-driven consistency training for semi-supervised head-and-neck tumor segmentation”, *Knowledge-Based Systems* **272**, 110598, URL <https://www.sciencedirect.com/science/article/pii/S0950705123003489> (2023).
- Silberman, N. and R. Fergus, “Indoor scene segmentation using a structured light sensor”, in “2011 IEEE international conference on computer vision workshops (ICCV workshops)”, pp. 601–608 (IEEE, 2011).
- Sime, D. M., G. Wang, Z. Zeng, W. Wang and B. Peng, “Semi-supervised defect segmentation with pairwise similarity map consistency and ensemble-based cross-pseudo labels”, *IEEE Transactions on Industrial Informatics* (2022).
- Smith, L. N. and N. Topin, “Super-convergence: Very fast training of neural networks using large learning rates”, (2018).
- Sohn, H., H. J. Lim, M. P. Desimio, K. Brown and M. Derriso, “Nonlinear ultrasonic wave modulation for online fatigue crack detection”, *Journal of Sound and Vibration* (2014).
- Song, K., X. Tan, T. Qin, J. Lu and T.-Y. Liu, “Mpnet: Masked and permuted pre-training for language understanding”, (2020).
- Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting”, *The journal of machine learning research* **15**, 1, 1929–1958, URL <http://jmlr.org/papers/v15/srivastava14a.html> (2014).
- Stahlschmidt, S., B. Ulfenborg and J. Synnergren, “Multimodal deep learning for biomedical data fusion: a review”, *Briefings in Bioinformatics* **23** (2022).
- Suankulova, K., “Correlation functions in determining the structure of chromosomes”, (2020).
- Sun, D., W. Zhu, X. Qiu, L. Liu, Y. Xiang and F.-Z. Xuan, “Nonlinear ultrasonic detection of closed cracks in metal plates with phase-velocity mismatching”, *NDT & E International* p. 102788 (2023).

- Suvdaa, B., J. Ahn and J. Ko, “Steel surface defects detection and classification using SIFT and voting strategy”, *International Journal of Software Engineering and its Applications* (2012).
- Tabernik, D., S. Šela, J. Skvarč and D. Skočaj, “Segmentation-based deep-learning approach for surface-defect detection”, *Journal of Intelligent Manufacturing* (2020).
- Tarvainen, A. and H. Valpola, “Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results”, in “Advances in Neural Information Processing Systems”, (2017).
- Te Han, Y.-F. L., “Out-of-distribution detection-assisted trustworthy machinery fault diagnosis approach with uncertainty-aware deep ensembles”, *Reliability Engineering and System Safety* (2022).
- Tippey, K. G., T. Roady, C. Rodriguez-Paras, L. J. Brown, W. G. Rantz and T. K. Ferris, “General aviation weather alerting: The effectiveness of different visual and tactile display characteristics in supporting weather-related decision making”, null (2017).
- Tohme, T., K. Vanslette and K. Youcef-Toumi, “Reliable neural networks for regression uncertainty estimation”, *Reliability Engineering and System Safety* (2022).
- Verdoja, F. and V. Kyrki, “Notes on the behavior of mc dropout”, (2021).
- Verma, V., A. Lamb, J. Kannala, Y. Bengio and D. Lopez-Paz, “Interpolation consistency training for semi-supervised learning”, in “IJCAI International Joint Conference on Artificial Intelligence”, (2019).
- Vivo, F. D., F. D. Vivo, E. N. Johnson and M. Battipede, “Infra-red line camera data-driven edge detector in uav forest fire monitoring”, *Aerospace Science and Technology* (2021).
- Wang, C., Y. Zhang, M. Cui, J. Liu, P. Ren, Y. Yang, X. Xie, X. Hua, H. Bao and W. Xu, “Active boundary loss for semantic segmentation”, *AAAI Conference on Artificial Intelligence* (2021a).
- Wang, C., Y. Zhang, M. Cui, J. Liu, P. Ren, Y. Yang, X. Xie, X. Hua, H. Bao and W. Xu, “Active boundary loss for semantic segmentation”, *CoRR* **abs/2102.02696**, URL <https://arxiv.org/abs/2102.02696> (2021b).
- Wang, G., J.-N. Hwang, C. Rose and F. Wallace, “Uncertainty-based active learning via sparse modeling for image classification”, *IEEE Transactions on Image Processing* **28**, 1, 316–329 (2018).

- Wang, W., F. Wei, L. Dong, H. Bao, N. Yang and M. Zhou, “Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers”, (2020a).
- Wang, Y., L. Gao, Y. Gao and X. Li, “A new graph-based semi-supervised method for surface defect classification”, *Robotics and Computer-Integrated Manufacturing* (2021c).
- Wang, Y., Y. Wang, Y. Gao, Y. Gao, Y. Gao, Y. Liu, Y. Liu, S. Ghosh, S. Ghosh, W. Subber, W. Subber, P. Pandita, P. Pandita, L. Wang and L. Wang, “Bayesian-entropy gaussian process for constrained metamodeling”, *Reliability Engineering and System Safety* (2021d).
- Wang, Y., Y. Wang and Y. Liu, “Bayesian entropy network for fusion of different types of information”, *Reliability Engineering and System Safety* (2020b).
- Whitehurst, G., L. J. Brown, W. G. Rantz, D. Nicolai and J. M. Bradley, “The effect of experiential education on pilots’ vfr into imc decision-making”, *Journal of Aviation/Aerospace Education /amp Research* (2019).
- Wilson, A. G. and P. Izmailov, “Bayesian deep learning and a probabilistic perspective of generalization”, *Advances in neural information processing systems* **33**, 4697–4708 (2020).
- Wilson, D. R. and T. A. Sloan, “Vfr flight into imc: Reducing the hazard”, null (2003).
- Woo, S., J. Park, J.-Y. Lee and I. S. Kweon, “Cbam: Convolutional block attention module”, in “Proceedings of the European conference on computer vision (ECCV)”, pp. 3–19 (2018).
- Woods, P. W. and P. D. Allen, “A cue generator for crack detection”, *Image and Vision Computing* (1989).
- Xu, H., H. Xiao, H. Hao, L. Dong, X. Qiu and C. Peng, “Semi-supervised learning with pseudo-negative labels for image classification”, *Knowledge-Based Systems* **260**, 110166, URL <https://www.sciencedirect.com/science/article/pii/S095070512201262X> (2023).
- Xu, Y., D. Li, Q. Xie, Q. Wu and J. Wang, “Automatic defect detection and segmentation of tunnel surface using modified mask r-cnn”, *Measurement* **178**, 109316 (2021).
- Yang, F., F. Yang, F. Yang, L. Zhang, L. Zhang, S. Yu, D. V. Prokhorov, X. Mei and H. Ling, “Feature pyramid and hierarchical boosting network for pavement crack detection”, *IEEE Transactions on Intelligent Transportation Systems* (2020a).

- Yang, F., L. Zhang, S. Yu, D. Prokhorov, X. Mei and H. Ling, “Feature pyramid and hierarchical boosting network for pavement crack detection”, *IEEE Transactions on Intelligent Transportation Systems* **21**, 4, 1525–1535 (2019).
- Yang, F., L. Zhang, S. Yu, D. Prokhorov, X. Mei and H. Ling, “Feature Pyramid and Hierarchical Boosting Network for Pavement Crack Detection”, *IEEE Transactions on Intelligent Transportation Systems* (2020b).
- Yang, M., P. Wu, J. Liu and H. Feng, “Memseg: A semi-supervised method for image surface defect detection using differences and commonalities”, *Engineering applications of artificial intelligence* (2022).
- Yang, W., W. Yang, L. Lorch, L. Lorch, M. A. Graule, M. A. Graule, H. Lakkaraju, H. Lakkaraju, F. Doshi-Velez and F. Doshi-Velez, “Incorporating interpretable output constraints in bayesian neural networks”, *arXiv: Learning* (2020c).
- Yang, Y. and M. Loog, “Active learning using uncertainty information”, in “2016 23rd International Conference on Pattern Recognition (ICPR)”, pp. 2646–2651 (IEEE, 2016).
- Yun, S., D. Han, S. Chun, S. J. Oh, J. Choe and Y. Yoo, “CutMix: Regularization strategy to train strong classifiers with localizable features”, in “Proceedings of the IEEE International Conference on Computer Vision”, (2019).
- Zhang, B. and X. Ma, “A review—pitting corrosion initiation investigated by tem”, *Journal of Materials Science & Technology* **35**, 7, 1455–1465 (2019).
- Zhang, C., J. Bütepage, H. Kjellström and S. Mandt, “Advances in variational inference”, *IEEE transactions on pattern analysis and machine intelligence* **41**, 8, 2008–2026 (2018).
- Zhang, H., M. Cisse, Y. N. Dauphin and D. Lopez-Paz, “Mixup: Beyond empirical risk minimization”, (2017a).
- Zhang, J., J. Xu, L. Zhu, K. Zhang, T. Liu, D. Wang and X. Wang, “An improved MobileNet-SSD algorithm for automatic defect detection on vehicle body paint”, *Multimedia Tools and Applications* (2020).
- Zhang, X. and S. Mahadevan, “Aircraft re-routing optimization and performance assessment under uncertainty”, *Decision Support Systems* (2017).
- Zhang, X. and S. Mahadevan, “Bayesian neural networks for flight trajectory prediction and safety assessment”, *Decision Support Systems* (2020).

- Zhang, Y., L. Yang, J. Chen, M. A. Fredericksen, D. P. Hughes and D. Z. Chen, “Deep adversarial networks for biomedical image segmentation utilizing unannotated images”, International Conference on Medical Image Computing and Computer-Assisted Intervention (2017b).
- Zhao, Z., R. Wu, G. Nie, B. Liu and X. Li, “A recognition method for multi-object information based on multi-source data fusion”, in “2021 6th International Conference on Robotics and Automation Engineering (ICRAE)”, pp. 376–380 (IEEE, 2021).
- Zhou, B., A. Khosla, A. Lapedriza, A. Oliva and A. Torralba, “Learning Deep Features for Discriminative Localization”, in “Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition”, (2016).
- Zhou, T., T. Han and E. L. Droguett, “Towards trustworthy machine fault diagnosis: A probabilistic bayesian deep learning framework”, Reliability Engineering & System Safety (2022).
- Zhu, R., Y. Chen, W. Peng and Z.-S. Ye, “Bayesian deep-learning for rul prediction: An active learning perspective”, Reliability Engineering & System Safety (2022).
- Zhu, W., R. Liang, J. Yang, Y. Cao, G. Fu and Y. Cao, “A sub-region unet for weak defects segmentation with global information and mask-aware loss”, Engineering applications of artificial intelligence (2023).
- Zhu, Y., N. Zabarlas, P.-S. Koutsourelakis and P. Perdikaris, “Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data”, Journal of Computational Physics **394**, 56–81 (2019).
- Zou, Q., Z. Zhang, Q. Li, X. Qi, Q. Wang and S. Wang, “Deepcrack: Learning hierarchical convolutional features for crack detection”, IEEE Transactions on Image Processing **28**, 3, 1498–1512 (2018).